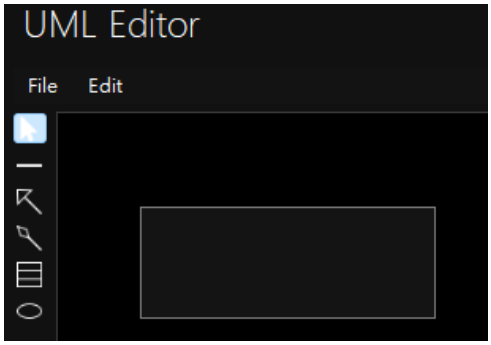
軟體工程實務 HW2

學號：113522118
姓名：韓志鴻

1. **Bug：**需求文件中提到「使用者不放開左鍵，進行拖曳(drag)的動作。畫布上會從滑鼠按下的起點至當前滑鼠的位置作為矩形的對角端點，繪製出一個矩形以提示使用者當前的選取範圍。」且「使用者拖曳到另外一個座標 x2, y2，放開左鍵 (mouse released)，矩形消失。」然而**實際操作發現僅能從左上至右下拖曳出矩形，若改變拖曳方向如從右上至左下，或從右下至左上拖曳，則無法繪製出矩形。**

   Solution：追蹤程式碼中 OnMouseDrag 函式(圖 2 紅框 if 部分)，發現過程使用 UpdateSelectedArea 函式(圖 3)，此函式定義拖曳矩形的左上點座標以及矩形長寬，但未考慮在其他拖曳方向下，宣告出的矩形長寬可能為負值。因此必須**修改成左上角點座標為拖曳起點與終點中(x, y)座標最小值，並且長寬距離計算要取絕對值**。



▲ 圖 1：在畫布上拖曳出矩形

```
public override void OnMouseDrag(object sender, MouseDragEventArgs e)
{
  if (_hasSelectArea)
  {
    //No shape selected means formed a selected area
    _currentMousePoint.X = e.X;
    _currentMousePoint.Y = e.Y;
    UpdateSelectedArea(Shape);
  }
  else
  {
    // 新增遞迴函式
    void MoveShapeAndChildren(Shape shape, int offsetX, int offsetY)
    {
      for (int i = 0; i < shape.Count; ++i) // 對於當前框起來內的所有物件
      {
        MoveShapeAndChildren(shape.GetShape(i), offsetX, offsetY); // 對於每個物件內部都要再DFS看是否還有Group起來的物件
      }

      shape.Move(offsetX, offsetY); // 移動遞迴到最底的每個物件和Group框
    }

    foreach (Shape shape in Canvas.SelectedShapes)
    {
      //shape.Move(e.OffsetX, e.OffsetY); 原寫法
      MoveShapeAndChildren(shape, e.OffsetX, e.OffsetY); // 新增 呼叫遞迴函式
    }
  }

  base.OnMouseDrag(sender, e);
}
```

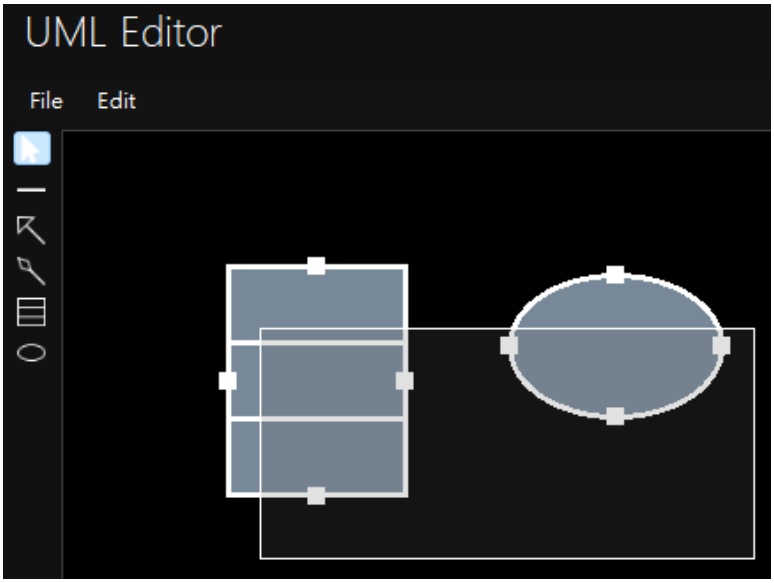
▲ 圖 2：OnMouseDrag 函式

```
private void UpdateSelectedArea(Shape selectedArea)
{
    int left = Math.Min(_mousePressedPoint.X, _currentMousePoint.X); // 原 _mousePressedPoint.X
    int top = Math.Min(_mousePressedPoint.Y, _currentMousePoint.Y); // 原 _mousePressedPoint.Y
    int width = Math.Abs(_currentMousePoint.X - _mousePressedPoint.X); // 原 _currentMousePoint.X - _mousePressedPoint.X
    int height = Math.Abs(_currentMousePoint.Y - _mousePressedPoint.Y); // 原 _currentMousePoint.Y - _mousePressedPoint.Y

    selectedArea.SetLocation(left, top);
    selectedArea.SetSize(width, height);
}
```

▲ 圖 3：UpdateSelectedArea 函式修改

2. **Bug：**對於已被 Group 的物件進行拖曳，只能拖曳最上層的 Group 物件。
   Solution：追蹤程式碼中 OnMouseDrag 函式(上圖 2 藍框 else 部分)，原本只有在 foreach 中 Move 一個 Shape，而該 Shape 就是最上層的 Group 物件，並沒有再往下去移動其他物件。故需要寫一個遞迴函式 MoveShapeAndChildren 進行 DFS，每次都先深度搜尋當前物件中是否還有其他物件，直到當前為最底物件時，Move 該物件並 return 回上一層，如此就能移動 Group 中的所有物件。



▲ 圖 4：矩形和橢圓形已組成 Group，但拖曳時僅能拖曳最上層的 Group 物件

3. **Bug：**對於被選取的連結線段按 Delete，會造成當機。
   Solution：追蹤 DeleteAction.cs 中的 Trigger 函式(圖 5)，發現 DestroyAllCombinations 函式內的 while 進入無限迴圈(圖 6)，再深入追蹤 while 內的 Destroy 函式(圖 7)發現，刪除的只有 Source 和 Destination 而沒有 Line，代表線段並沒有真正被刪除。因此要**按照 Source 和 Destination 的寫法，補上 Line.RemoveCombination(this)**，才能真正的刪除線段而不會造成無限迴圈。

```
public override void Trigger()
{
    foreach (Shape shape in Canvas.SelectedShapes)
    {
        shape.DestroyAllCombinations();
        Canvas.RemoveShape(shape);
    }

    base.Trigger();
}
```

▲ 圖 5：Trigger 函式

```
public virtual void DestroyAllCombinations()
{
    Debug.Assert(Combinations != null);

    while (Combinations.Count > 0)
    {
        Combinations[0].Destroy();
    }
}
```

▲ 圖 6：DestroyAllCombinations()內的 while 會出現無限迴圈

```
public void Destroy()
{
    if (Source != null)
    {
        Source.RemoveCombination(this);
    }

    if (Destination != null)
    {
        Destination.RemoveCombination(this);
    }

    if (Line != null)
    {
        Line.IsSelected = false;
        Line.RemoveCombination(this); // 新增
        Canvas.GetInstance().RemoveShape(Line);
    }
}
```

▲ 圖 7：程式碼修改，補上 Line.RemoveCombination(this)