

# HW2\_109403021

## 1. Bug1: delete 任何連結物件(connection line)都會當掉

### Debug 流程:

我先到 Actions 中的 DeleteActions.cs 看看

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

    base.Trigger();
}
```

結果一路追蹤到 Shape.cs 中發現會在下圖處陷入無限迴圈，  
Combinations.Count 值始終會是 1

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

    while (Combinations.Count > 0)
    {
        已歷時 ≤ 1 毫秒
        Combinations[0].Destroy();
    }
}
```

| 監看式 1                 |           |
|-----------------------|-----------|
| 搜尋 (Ctrl+E) 🔍 搜尋深度: 3 |           |
| 名稱                    | 值         |
| Combinations          | Count = 1 |
| 新增要監看的項目              |           |

所以接著到 Destroy()裡面看看

```
public void Destroy()
{
    已歷時 ≤ 1 毫秒
    if (Source != null)
    {
        Source.RemoveCombination(this);
    }

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

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

看起來就是把 Source, Destination, Line 做 Remove，而問題出在 Line 上 Line 的判斷中相比上面兩個又缺少調用 RemoveCombination，所以我先嘗試加入後修改如下

```
public void Destroy()
{
    已歷時 ≤ 1 毫秒
    if (Source != null)
    {
        Source.RemoveCombination(this);
    }

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

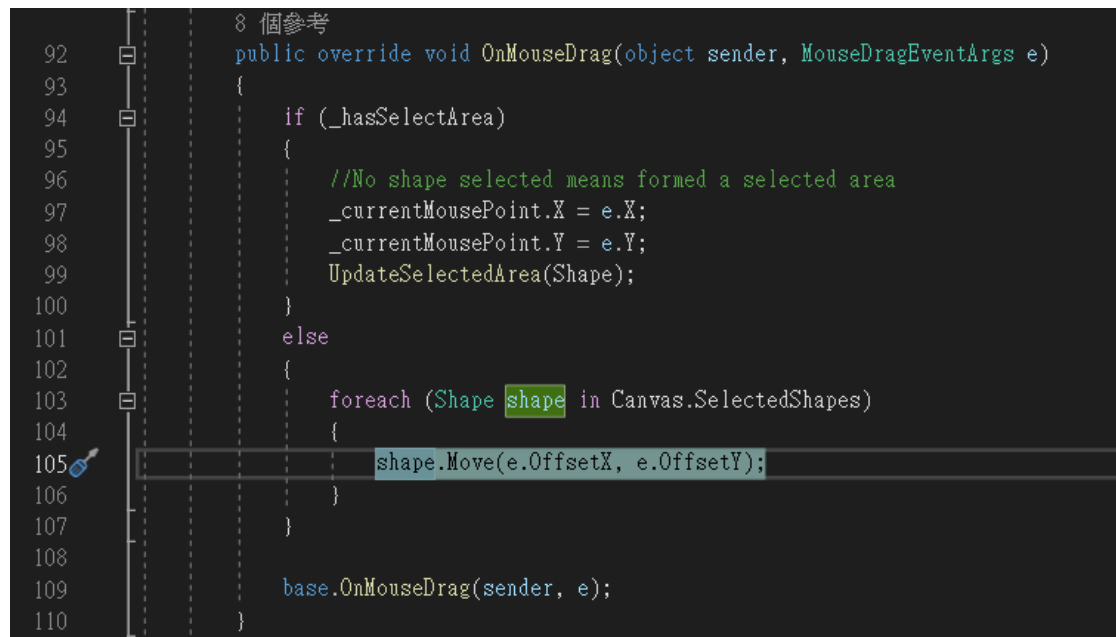
    if (Line != null)
    {
        Line.IsSelected = false;
        Canvas.GetInstance().RemoveShape(Line);
        Line.RemoveCombination(this);
    }
}
```

結果加這一行就直接能成功刪除連結物件(connection line)了

## 2. Bug2: Group 無法成功移動，只有 Group 的大框框會動，子物件停留在原地

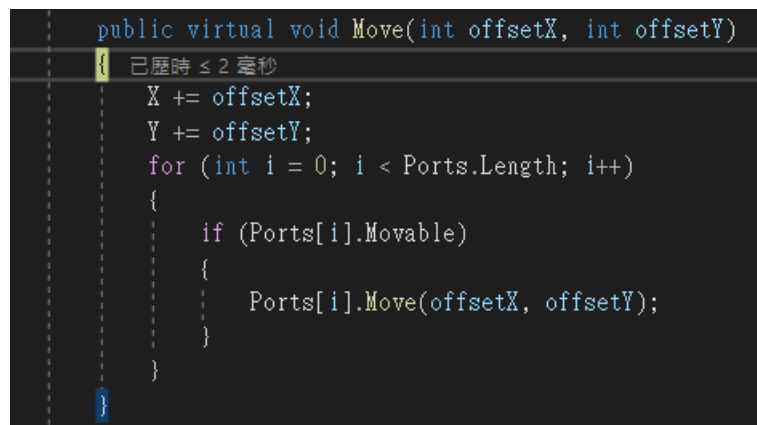
### Debug 流程:

由於移動是在 **SelectMode** 下執行，所以我先到 **SelectMode.cs** 中找有沒有符合移動動作的 **function**，由於移動會有 **drag** 的動作，我找到看似符合的如下



```
8 個參考
92 public override void OnMouseDown(object sender, MouseEventArgs e)
93 {
94     if (_hasSelectArea)
95     {
96         //No shape selected means formed a selected area
97         _currentMousePoint.X = e.X;
98         _currentMousePoint.Y = e.Y;
99         UpdateSelectedArea(Shape);
100     }
101     else
102     {
103         foreach (Shape shape in Canvas.SelectedShapes)
104         {
105             shape.Move(e.OffsetX, e.OffsetY);
106         }
107     }
108     base.OnMouseDown(sender, e);
109 }
110
```

從上圖第 105 行進入到 **Shape.cs** 的 **Move** 函式中如下



```
public virtual void Move(int offsetX, int offsetY)
{
    已歷時 ≤ 2 毫秒
    X += offsetX;
    Y += offsetY;
    for (int i = 0; i < Ports.Length; i++)
    {
        if (Ports[i].Movable)
        {
            Ports[i].Move(offsetX, offsetY);
        }
    }
}
```

可以看到只有移動物件本身還有調整端口的程式碼，沒有動到子物件，所以子物件會原地停留，於是我嘗試加入移動子物件的程式碼，首先找到子物件存放在物件哪裡

```

133     /// </summary>
134     protected List<Shape> Shapes;
135

```

|         |                                       |    |                        |
|---------|---------------------------------------|----|------------------------|
| Ports   | {UMLEditor.Pseudo.Port[4]}            | 檢視 | UMLEditor.Pseudo.P...  |
| Shapes  | Count = 2                             | 檢視 | System.Collections.... |
| [0]     | {UMLEditor.Shapes.BasicObjects.Class} |    | UMLEditor.Shapes.S...  |
| [1]     | {UMLEditor.Shapes.BasicObjects.Class} |    | UMLEditor.Shapes.S...  |
| 未經處理的檢視 |                                       |    |                        |
| Width   | 343                                   |    | int                    |
| Y       | 89                                    |    | int                    |

了解好 Shape 這個物件的型態後修改程式碼如下

```

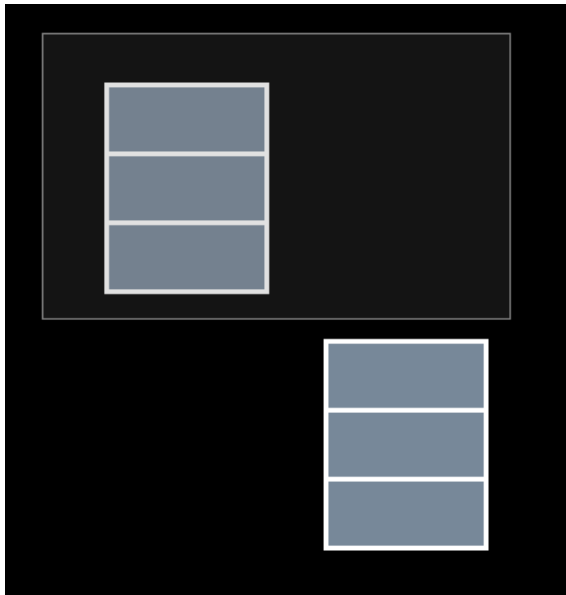
public virtual void Move(int offsetX, int offsetY)
{
    X += offsetX;
    Y += offsetY;
    if (Shapes != null)
    {
        for (int i = 0; i < Shapes.Count; i++)
        {
            Shapes[i].Move(offsetX, offsetY);
        }
    }

    for (int i = 0; i < Ports.Length; i++)
    {
        if (Ports[i].Movable)
        {
            Ports[i].Move(offsetX, offsetY);
        }
    }
}

```

然後就能成功將整個 Group 一同移動

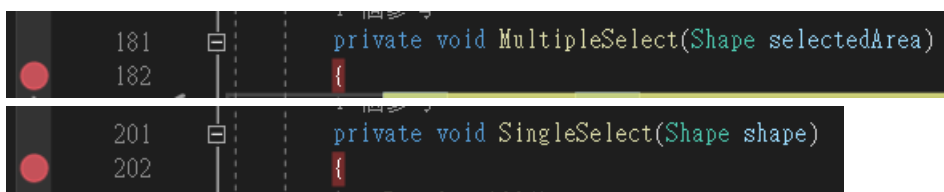
### 3. Bug3: 有時候選取單個物件其他物件也會被選到



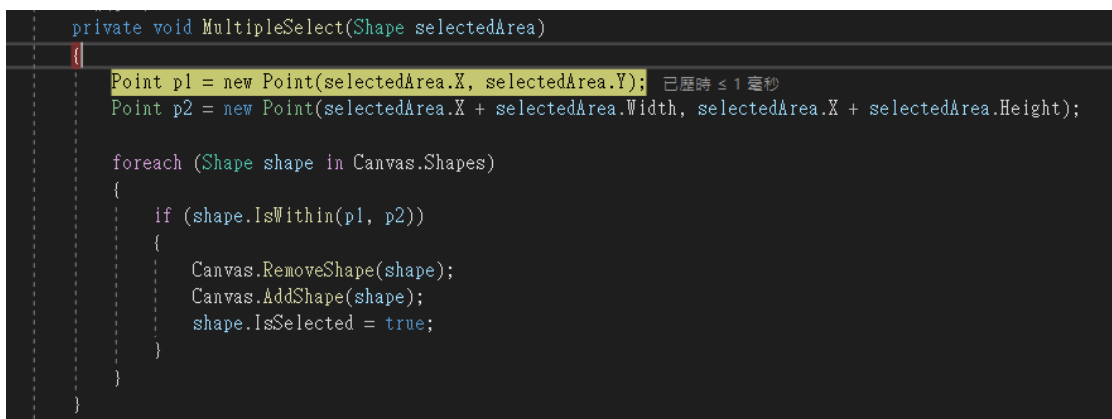
像左圖的選取範圍誤使兩物件都被選取

#### Debug 流程:

由於也是在 `SelectMode` 下發生的 Bug，一樣先從 `SelectModes.cs` 中看看，看到兩個應該和選取有關的 `function`，所以我在兩者都設中斷點



然後在 GUI 中拉出一個選取範圍，結果進入的是 `MultipleSelect`



看起來是會以拉出的範圍設置兩點 `p1,p2` 並將兩點間的範圍的 `shape` 物件 `IsSelect` 設為 `true`

| 區域變數                                  |  |                       |
|---------------------------------------|--|-----------------------|
| 名稱                                    | 值  | 類型                    |
| UMLEditor.Shapes.Shape.Height.get ... | 179  | int                   |
| ▶ this                                | {UMLEditor.Modes.SelectMode}                 | UMLEditor.Modes.S...  |
| ▶ selectedArea                        | {UMLEditor.Shapes.BasicObjects.SelectedArea} | UMLEditor.Shapes.S... |
| ▶ p1                                  | {X = 328 Y = 62}                             | System.Drawing.Poin   |
| ▶ p2                                  | {X = 602 Y = 507}                            | System.Drawing.Poin   |

602-328=274, 507-62=445，所以 p1,p2 會形成一個 width:274 height:445 的選取範圍，但我的選取範圍應該是 width 比 height 大的如下圖



所以可以知道 p1,p2 設置時就錯了，因此修改如下

```
private void MultipleSelect(Shape selectedArea)
{
    Point p1 = new Point(selectedArea.X, selectedArea.Y);
    Point p2 = new Point(selectedArea.X + selectedArea.Width, selectedArea.Y + selectedArea.Height);

    foreach (Shape shape in Canvas.Shapes) 已歷時 ≤ 2 毫秒
    {
        if (shape.IsWithin(p1, p2))
        {
            Canvas.RemoveShape(shape);
            Canvas.AddShape(shape);
            shape.IsSelected = true;
        }
    }
}
```

原來是 p2 的 y 座標設置錯誤，修改後即解決 bug3