DoubleSpongeBob_3_Scaling_Translation =>
    DoubleSpongeBob_4_Scaling_Translation_Synch

1. Add DrawingListener interface

[DrawingListener.java]

```
public interface DrawingListener {

    void translationChanged(int w_newTranslateX, int w_newTranslateY);
}
```

2. Add list of listeners and addDrawingListener method to DrawingComponent

```
private ArrayList<DrawingListener> listeners;

public void addDrawingListener(DrawingListener listener) {
    listeners.add(listener);
}
```

3. Add notifyTranslationChanged method to DrawingComponent and call
    it from mouseDragged

```
private void notifyTranslationChanged(int w_newTranslateX, int w_newTranslateY) {
    for (DrawingListener listener : listeners) {
        listener.translationChanged(w_newTranslateX, w_newTranslateY);
    }
}

@Override
public void mouseDragged(MouseEvent e) {
    if (dragging) {
        int d_X = e.getX();
        int d_Y = e.getY();

        AffineTransform transform = new AffineTransform();
        transform.scale(scale, scale);
        transform.translate(w_dragStartTranslateX, w_dragStartTranslateY);

        Point2D d_Pt = new Point2D.Double(d_X, d_Y);
        Point2D w_Pt = new Point2D.Double();
        try
        {
            transform.inverseTransform(d_Pt, w_Pt);
        }
        catch (NoninvertibleTransformException ex) {
            return;
        }
        int w_X = (int)w_Pt.getX();
        int w_Y = (int)w_Pt.getY();
```

```
        int w_deltaX = w_X - w_dragStartX;
        int w_deltaY = w_Y - w_dragStartY;

        w_translateX = w_dragStartTranslateX + w_deltaX;
        w_translateY = w_dragStartTranslateY + w_deltaY;

*****       notifyTranslationChanged(w_translateX, w_translateY);

        repaint();
    }
}
```

4. Implement two listeners on the Drawing class (one for each window),
   add addDrawingListener method to DrawingFrame, and
   add a listener to each frame in Drawing

[Drawing]

```
private DrawingListener drawingListener1 = new DrawingListener() {

    @Override
    public void translationChanged(int w_newTranslateX, int w_newTranslateY) {
        frame2.setTranslation(w_newTranslateX, w_newTranslateY);
    }
};

private DrawingListener drawingListener2 = new DrawingListener() {

    @Override
    public void translationChanged(int w_newTranslateX, int w_newTranslateY) {
        frame1.setTranslation(w_newTranslateX, w_newTranslateY);
    }
};
```

[Drawing.run]

```
frame1.addDrawingListener(drawingListener1);
frame2.addDrawingListener(drawingListener2);
```

[DrawingFrame]

```
public void addDrawingListener(DrawingListener listener) {
    component.addDrawingListener(listener);
}
```

5. Add setTranslate methods to DrawingComponent and DrawingFrame

[DrawingComponent]

```
public void setTranslation(int w_newTranslateX, int w_newTranslateY) {
    w_translateX = w_newTranslateX;
```

```
    w_translateY = w_newTranslateY;
    this.repaint();
}

[DrawingFrame]

public void setTranslation(int w_newTranslateX, int w_newTranslateY) {
    component.setTranslation(w_newTranslateX, w_newTranslateY);
}
```