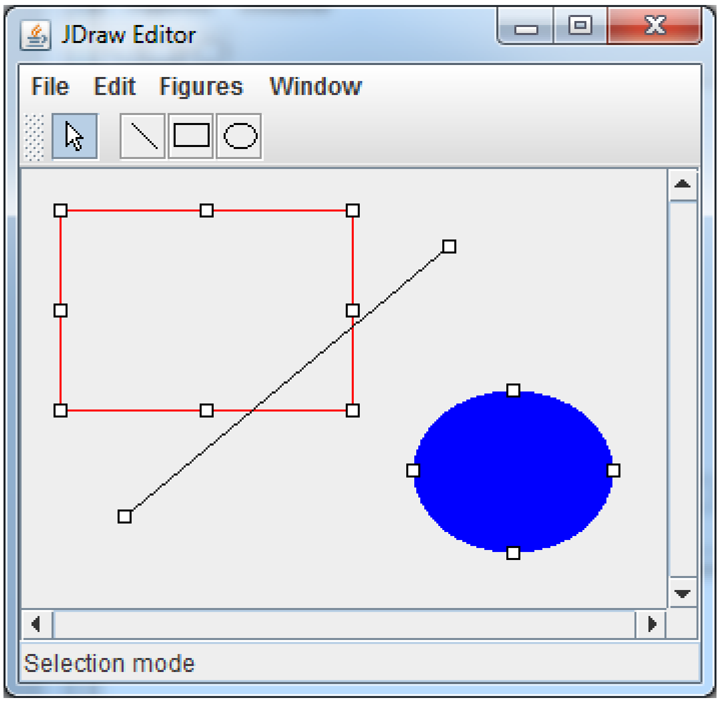
Assignment 4: Figure Handles

With the last two assignments you broke into the JDraw framework, you implemented the model, you extended the editor with your own figures and you implemented the updating of the views so that all windows always show a consistent view of the model. Some methods defined in the interface *Figure* have not yet been implemented. With this assignment you will implement method *getHandles.*

Method *getHandles* returns the handles of a figure. These handles are used to change the shape of a figure. In the picture below the line provides two, the oval four and the rectangle eight handles.



The handles implement the state pattern, because depending on the handle type (N, E, S, W, etc) the behavior of the handle is slightly different.

Handles are objects that implement the interface *jdraw.framework.FigureHandle*. Similar to the Figure interface, this interface provides methods *draw, contains* and *getBounds*.

Method *getCursor* defines the cursor to be used when the mouse is over a figure handle. This method is called by the framework.

Method *getOwner* returns a reference to the corresponding figure the handle belongs to.

If the mouse is pressed inside a handle, the methods *startInteraction*, *dragInteraction* and *stopInteraction* are called on that handle. As a consequence, the corresponding figure will be adjusted, typically using method *setBounds* on the figure. This method sends figure events upon a change of the figure which are consumed by the model, therefore an explicit call of the repaint method from within the handle is not necessary. The additional *MouseEvent* parameter can be used to read the modifier keys, and the *DrawView* parameter can be used to access the *DrawContext*, e.g. to display messages in the status bar.

Consider (as in Assignment 3) where abstract base classes could be introduced.

The JDraw project that we provided is also prepared for the definition and execution of JUnit tests. In class *jdraw.test.JDrawTests* two test classes are combined into a test suite. Extend this class during your development (or better before your development) with tests for your classes.

Optional extension:
Extend the handles so that the modification of the figure depends on whether modifier keys are pressed while dragging the handles. If the Ctrl-key is pressed, then the center of the figure could remain fixed, and if the Shift-key is pressed then the ratio of the width and the height of the figure could remain fixed. This behavior depends on the state of the modifier keys and thus could be implemented with the help of the state pattern again

Deadline: October 23, 2018