Playground 1/4

```
import java.awt.Frame;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

class PlayGround extends Frame implements ActionListener {
  private static final int     DIMENSION              = 3;
  private static final int     NR_IN_ROW_FOR_SUCCESS = 3;
  private static final boolean SINK_DOWN              = false;

  private final int height;
  private final int width;
  private       int dimension;
  private Figure [][] fig;
  private Evaluator eval;
  private boolean sinkDown;
```

Playground 2/4

```
public PlayGround ( int dimension, Evaluator eval, boolean sinkDown ) {
  super( "Tic Tac Toe" );
  height = dimension * 100;
  width  = dimension * 100;
  this.dimension = dimension;
  fig = new Figure[dimension][dimension];
  this.eval = eval;
  this.sinkDown = sinkDown;
  setSize( width, height );
  setLayout( new GridLayout( dimension, dimension ) );
  for ( int i = 0; i < dimension; i++ )
      for ( int j = 0; j < dimension; j++ ) {
          fig[i][j] = new Figure();
          add( fig[i][j] );
          fig[i][j].addActionListener( this );
      }

  addWindowListener( new WindowAdapter() {
      public void windowClosing ( WindowEvent e ) {
          System.exit(0);
      }
  });
}
```

Playground 3/4

```
public void actionPerformed( ActionEvent e ) {
  int row;
  for ( int i = 0; i < dimension; i++ )
    for ( int j = 0; j < dimension; j++ )
      if ( e.getSource() == fig[i][j] ) {
        if ( sinkDown ) {
          row = 0;
          while ( (row < dimension) && (fig[row][j].symbol == Figure.NONE) )
            row++;
          row--;
          if ( (row >= 0) && (fig[row][j].symbol == Figure.NONE) )
            setFigure( row, j );
        } else if ( fig[i][j].symbol == Figure.NONE ) {
          setFigure( i, j );
        }
      }
} // actionPerformed
```

Playground 4/4

```
  private void setFigure( int i, int j ) {
    fig[i][j].symbol = Figure.currentSymbol;
    fig[i][j].setText( new Character(Figure.currentSymbol).toString() );
    if ( eval.succeeded( fig, Figure.currentSymbol ) ) {
      for ( int i1 = 0; i1 < dimension; i1++ )
        for ( int j1 = 0; j1 < dimension; j1++ )
          fig[i1][j1].removeActionListener( this );
      new SuccessFrame( Figure.currentSymbol, this );
    } else if ( eval.undecided( fig, Figure.NONE ) ) {
      new UndecidedFrame( this );
    }
    Figure.toggleSymbol();
  }

  public static void main(String[] args) {
    Evaluator  eval = new Evaluator( DIMENSION, NR_IN_ROW_FOR_SUCCESS );
    PlayGround field = new PlayGround( DIMENSION, eval, SINK_DOWN );
    field.setVisible(true);
  }

}
```