5.3.2020 10:44:10 **ShipNavigationSystem.java** Page 1/1

```
//=============================================================================
// Projekt      : MAS-SE 20: Java-Vorkurs
// Titel        : Ü31: Schiff-Navigations-System: Ausgangslage
// Dateiname    : ShipNavigationSystem.java
// Autor        : 'You'
// Beschreibung : Ein 'Spiel', bei welchem Schiffe in einer X/Y-Ebene positioniert
//                werden können und man daraufhin diese Schiffe an neue
//                X/Y-Koordinaten fahren lassen kann ;-)
//=============================================================================

import java.awt.Toolkit;
import java.util.ArrayList;

public class ShipNavigationSystem {

    private final static ArrayList<Ship> mShips = new ArrayList<Ship>();
    private final static Screen mScreen = new Screen(30, 10);


    public static void main(String[] args) {

        // TODO: Implement here ...

    }

    // TODO: Implement here ...

}
```

5.3.2020 10:43:59 **Screen.java** Page 1/5

```
import java.util.Scanner;

/**
 * Abstracts a Screen with a Field of Rows and Columns (ROWS * COLS),
 * a Menu-Area (right of the Field) and an User-Input-Area (below the Menu-Area).
 */
public class Screen {

    /**
     * Number of Columns in the 'Field'.
     */
    public final int COLS;

    /**
     * Number of Rows in the 'Field'.
     */
    public final int ROWS;

    /**
     * Number of Colums/Rows of the Border of the 'Field'.
     */
    private final int BORDER = 3;

    /**
     * Distance from Border to Menu.
     */
    private final int MENU_X_OFFSET = 10;

    /**
     * Total X-Length of the Screen.
     */
    private final int X_LENGTH = 79;

    /**
     * Total two-dimensional Char-Array which represents the whole Screen.
     * First-Dimension  : Y-Direction (Rows)
     * Second-Dimension : X-Direction (Columns)
     */
    private char[][] mScreen;

    private Scanner mScanner;


    /**
     * Initializes the whole Screen (incl. the Field of Rows and Columns with
     * the Borders).
     * @param pCols The Number of Columns in the Field.
     * @param pRows The Number of Rows in the Field.
     */
    public Screen(int pCols, int pRows) {
        // TODO: Implement here ...
    }


    private void writeFieldBorders() {
        for (int col = BORDER - 1; col < COLS + BORDER + 1; col++) {
            mScreen[BORDER - 1][col] = mScreen[ROWS + BORDER][col] = '-';
        }
        for (int row = BORDER - 1; row < ROWS + BORDER + 1; row++) {
            mScreen[row][BORDER - 1] = mScreen[row][COLS + BORDER] = '|';
        }
        mScreen[BORDER - 1][BORDER - 1] = mScreen[BORDER - 1][COLS + BORDER]
            = mScreen[ROWS + BORDER][BORDER - 1]
            = mScreen[ROWS + BORDER][COLS + BORDER] = '+';

        for (int col = 0; col < COLS; col++) {
            if (((col + 1) % 10) == 0) {
                mScreen[0][BORDER + col] = (char) ('0' + ((col + 1) / 10));
                mScreen[BORDER + ROWS + BORDER - 1][BORDER + col]
                        = (char) ('0' + ((col + 1) / 10));
            }
```

```
72          }
73          for (int col = 0; col < COLS; col++) {
74              mScreen[1][BORDER + col] = (char) ('0' + ((col + 1) % 10));
75              mScreen[BORDER + ROWS + BORDER - 2][BORDER + col]
76                  = (char) ('0' + ((col + 1) % 10));
77          }
78          for (int row = 0; row < ROWS; row++) {
79              if (((row + 1) % 10) == 0) {
80                  mScreen[BORDER + row][0] = (char) ('0' + ((row + 1) / 10));
81                  mScreen[BORDER + row][BORDER + COLS + BORDER - 2]
82                      = (char) ('0' + ((row + 1) / 10));
83              }
84          }
85          for (int row = 0; row < ROWS; row++) {
86              mScreen[BORDER + row][1] = (char) ('0' + ((row + 1) % 10));
87              mScreen[BORDER + row][BORDER + COLS + BORDER - 1]
88                  = (char) ('0' + ((row + 1) % 10));
89          }
90      }
91
92
93      /**
94       * Sets the defined Position with a new Value.
95       * @param pX The X-Position (Column).
96       * @param pY The Y-Position (Row).
97       * @param pNewValue The new Value to be set at Position(x/y).
98       */
99      public void set(int pX, int pY, char pNewValue) {
100         // TODO: Implement here ...
101     }
102
103
104     /**
105      * Returns the Value of the defined Position.
106      * @param pX The X-Position (Column).
107      * @param pY The Y-Position (Row).
108      * @return The Value at Position(x/y).
109      */
110     public char get(int pX, int pY) {
111         return 0; // TODO: Implement here ...
112     }
113
114
115     /**
116      * Writes the String-Array pMenu to the Menu-Area.
117      * @param pMenu
118      */
119     public void setMenu(String[] pMenu) {
120         int xOffset = BORDER+COLS+BORDER+MENU_X_OFFSET;
121         int menuLineLength = mScreen[0].length - xOffset;
122         char[] menuEmptyLine = new char[menuLineLength];
123         for (int i = 0; i < menuLineLength; i++) {
124             menuEmptyLine[i] = ' ';
125         }
126         for (int row = 0; row < BORDER+ROWS+BORDER; row++) {
127             System.arraycopy(menuEmptyLine, 0, mScreen[row],
128                              xOffset, menuEmptyLine.length);
129         }
130         int yOffset = BORDER+ROWS+BORDER - pMenu.length - 1;
131         char[] menuTitle = "Menu".toCharArray();
132         System.arraycopy(menuTitle, 0, mScreen[yOffset-2],
133                          xOffset, menuTitle.length);
134         char[] underline = "----".toCharArray();
135         System.arraycopy(underline, 0, mScreen[yOffset-1],
136                          xOffset, underline.length);
137         for (int i = 0; i < pMenu.length; i++) {
138             char[] menuArr = pMenu[i].toCharArray();
139             System.arraycopy(menuArr, 0, mScreen[yOffset+i],
140                              xOffset, menuArr.length);
141         }
142     }
```



```
143
144
145     /**
146      * Reads an User-Input with a Prompt in the User-Input-Area (below Menu-Area).
147      *
148      * @param pPrompt The Prompt-String to be used.
149      * @return The Input-String given by the User.
150      */
151     public String userInput(String pPrompt) {
152         return null; // TODO: Implement here ...
153     }
154
155
156     /**
157      * Writes the Content of the 'Screen' to the Console.
158      */
159     public void print() {
160         System.out.println("\n\n\n\n");
161         for (int y = 0; y < mScreen.length; y++) {
162             System.out.println(mScreen[y]);
163         }
164         System.out.println("\n\n");
165     }
166
```

```
167
168     public static void main(String[] args) {
169
170         Screen screen = new Screen(30, 10);
171
172         screen.set(10, 2, 'X');
173         screen.set(20, 8, 'Y');
174         final String[] MENU = {
175                 "1. Neues Schiff",
176                 "2. Neue Position",
177                 "3. Ende"
178         };
179         screen.setMenu(MENU);
180
181         screen.print();
182
183         String input = screen.userInput("Ihre Wahl: ");
184         System.out.println("User-Input: >" + input + "<\n");
185
186         System.out.println("screen.get(10, 2): '" + screen.get(10, 2)+"'");
187         System.out.println("screen.get( 1, 1): '" + screen.get( 1, 1)+"'");
188     }
189 }
190
191 /* Session-Log:
192
193             1         2         3
194    123456789012345678901234567890
195   +------------------------------+
196  1|                              | 1
197  2|         X                    | 2
198  3|                              | 3
199  4|                              | 4
200  5|                              | 5
201  6|                              | 6
202  7|                              | 7
203  8|                    Y         | 8          Menu
204  9|                              | 9          ----
205 10|                              |10          1. Neues Schiff
206   +------------------------------+            2. Neue Position
207    123456789012345678901234567890             3. Ende
208             1         2         3
209
210
211
212                                                Ihre Wahl: 1
213 User-Input: >1<
214
215 screen.get(10, 2): 'X'
216 screen.get( 1, 1): ' '
217
218 */
219
220
```



```
221
```

```
1
2  public class Ship {
3
4      /**
5       * Ship is in normal Condition (not sunken).
6       */
7      public static final int OK = 1;
8      /**
9       * Ship is sunken.
10      */
11     public static final int SUNKEN = 2;
12
13     // TODO: Implement here ...
14
15
16     public Ship(int pXPos, int pYPos, String pName) {
17         // TODO: Implement here ...
18     }
19
20     public int getXPos() {
21         // TODO: Implement here ...
22     }
23
24     public void setXPos(int pXPos) {
25         // TODO: Implement here ...
26     }
27
28     public int getYPos() {
29         // TODO: Implement here ...
30     }
31
32     public void setYPos(int pYPos) {
33         // TODO: Implement here ...
34     }
35
36     public String getName() {
37         // TODO: Implement here ...
38     }
39
40     public void setName(String pName) {
41         // TODO: Implement here ...
42     }
43
44     public int getState() {
45         // TODO: Implement here ...
46     }
47
48     public void setState(int pState) {
49         // TODO: Implement here ...
50     }
51
52     public void print() {
53         // TODO: Implement here ...
54     }
55
56
```



```
56
57     public static void main(String[] args) {
58         Ship ship = new Ship(4, 2, "Santa-Maria");
59         ship.print();
60         ship.setState(Ship.SUNKEN);
61         ship.print();
62     }
63
64 }
65
66 /* Session-Log:
67
68 Ship:
69   XPos  = 4
70   YPos  = 2
71   Name  = Santa-Maria
72   State = OK
73 Ship:
74   XPos  = 4
75   YPos  = 2
76   Name  = Santa-Maria
77   State = SUNKEN
78
79 */
```