```
/* TTT.java                                  *
/* Hauptklasse fuer Tic Tac Toe mit Java AWT *
/* benoetigt Java 1.2 oder hoeher            *
/*                                           *
/* (c) Thomas Duebendorfer, Juli 2000        *
/* Fuer Lehrzwecke frei verwendbar und anpassbar.  *
/* Kommerzielle Verwendung verboten.         *
/*                                           */

import java.awt.*;
import java.awt.event.*;

public class TTT extends Frame implements ActionListener {
    //MenuItems deklarieren (fuer spaetere Bestimmung der Ereignis-Quell
    private MenuItem neu, laden, speichern, beenden;
    private MenuItem ueber;
    // NeuesSpielButton deklariere
    private Button neuesSpielButton;
    // DropDownListe fuer Spielerauswahl deklariere
    private Choice spielerChoice;

    // Die Felder sind von links oben nach rechts unten mit 0 bis 8 nummerie
    public int[] spielfeld;
    // Reihenfolge der Spielfelder (Arrayindizes
    //|0|1|2|
    //|3|4|5|
    //|6|7|8|
    // Codierung der Spielfeldeintraeg
    public static final int SPIELER_X = 10, SPIELER_O = 1, LEER = 0, UNENTSCHIEDEN = -1,
UNKLAR = 0;
    public int aktuellerSpieler, computerSpieler;
    public int anzahlGespielteZuege;

    public final static int BEREIT = 0;
    public final static int FERTIG = 1;
    public int status;

    public TTTBoard tttboard;

    TTT(String s) {
        // Konstruktor
        super(s);
        // Initialisierungen von Instanzvariable
        spielfeld = new int[9];

        // GUI aufbauen - LayoutManager definiere
        setLayout(new BorderLayout());

        // Menu definierer
        // MenuBar erzeugen, Menus und MenuItems erzeuge
        MenuBar mb = new MenuBar();
        Menu spiel = new Menu("Spiel");
        neu       = new MenuItem("Neu");
        laden     = new MenuItem("Laden ...");
        speichern = new MenuItem("Speichern ...");
        beenden   = new MenuItem("Beenden");

        Menu hilfe = new Menu("?");
        ueber = new MenuItem("Über ...");

        //Ereignisbehandlung hinzufuegen zu MenuItem
        neu.addActionListener(this);
        laden.addActionListener(this);
```



```
        neu.addActionListener(this);
        speichern.addActionListener(this);
        beenden.addActionListener(this);
        ueber.addActionListener(this);

        // Menupunkte zu Menus hinzufuege
        spiel.add(neu);
        spiel.addSeparator();
        spiel.add(laden);
        spiel.add(speichern);
        spiel.addSeparator();
        spiel.add(beenden);

        hilfe.add(ueber);

        mb.add(spiel);
        mb.add(hilfe);
        setMenuBar(mb);

        // Board definierer
        tttboard = new TTTBoard(this);
        add(tttboard, BorderLayout.CENTER);

        // DropDown-Liste, Neues-Spiel-Button definiere
        Panel horizPanel = new Panel(new BorderLayout());
        neuesSpielButton = new Button("Neues Spiel");

        // Ereignisbehandlung fuer NeuesSpielButto
        neuesSpielButton.addActionListener(this);
        horizPanel.add(neuesSpielButton, BorderLayout.EAST);

        spielerChoice = new Choice();
        spielerChoice.add("X beginnt");
        spielerChoice.add("O beginnt");
        spielerChoice.add("X beginnt gegen Computer");
        spielerChoice.add("O beginnt gegen Computer");
        horizPanel.add(spielerChoice, BorderLayout.WEST);
        add(horizPanel, BorderLayout.SOUTH);

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

    public static void main(String argv[]) {
        TTT ttt = new TTT("Tic Tac Toe");
        ttt.setSize(300, 350);
        ttt.neuesSpiel();
        ttt.setVisible(true);
    }

    public int getAktuellerSpieler() {
        // gibt aktuellen Spieler zuruec
        return aktuellerSpieler;
    }

    public int getGegner(int aktuellerSpieler) {
        // gibt den Gegner zu 'aktuellerSpieler' zuruec
        if (aktuellerSpieler == SPIELER_X)
            return SPIELER_O;
        else
```

```
126:                return SPIELER_X;
127:        }
128:
129:        public void neuesSpiel() {
130:            // Initialisierungen, um ein neues Spiel beginnen zu koenne
131:            computerSpieler = 0; // kein ComputerSpieler als Standar
132:            anzahlGespielteZuege = 0;
133:            loescheSpielfeld();
134:            // Startspieler aus DropDown auslese
135:            int aktuellerSpielerIndex = spielerChoice.getSelectedIndex();
136:            if (aktuellerSpielerIndex == 0)      { aktuellerSpieler = SPIELER_X; }
137:            else if (aktuellerSpielerIndex == 1) { aktuellerSpieler = SPIELER_O; }
138:            else if (aktuellerSpielerIndex == 2) { aktuellerSpieler = SPIELER_X;
        computerSpieler = SPIELER_O; }
139:            else if (aktuellerSpielerIndex == 3) { aktuellerSpieler = SPIELER_O;
        computerSpieler = SPIELER_X; }
140:            status = BEREIT;
141:        }
142:
143:        public void loescheSpielfeld() {
144:            // ganzes Spielfeld (inkl. GUI-Board) loesche
145:            for (int i = 0; i < 9; i++) {
146:                spielfeld[i] = 0;
147:                tttboard.setzeFeld(i,LEER);
148:            }
149:        }
150:
151:        public int istSpielFertig() {
152:            // gibt Gewinner SPIELER_X  bzw. SPIELER_O zuruec
153:            // oder UNENTSCHIEDEN, falls UNENTSCHIEDEN is
154:            // oder UNKLAR, falls Spiel noch nicht FERTIG is
155:            int summe;
156:
157:            // pruefe, ob es 3 gleiche Symbole in einer Zeile ha
158:            for (int i=0; i<=6; i = i + 3) {
159:                summe = spielfeld[i] +  spielfeld[i+1] +  spielfeld[i+2];
160:                if ((summe == 3) | (summe == 30))
161:                    return summe/3;
162:            }
163:
164:            // pruefe, ob es 3 gleiche Symbole in einer Spalte hat
165:            for (int i=0; i<3; i++) {
166:                summe = spielfeld[i] +  spielfeld[i+3] +  spielfeld[i+6];
167:                if ((summe == 3) | (summe == 30))
168:                    return summe/3;
169:            }
170:
171:            // pruefe, ob es 3 gleiche Symbole in einer Diagonale hat
172:            summe = spielfeld[0] +  spielfeld[4] +  spielfeld[8];
173:            if ((summe == 3) | (summe == 30))
174:                    return summe/3;
175:            summe = spielfeld[6] +  spielfeld[4] +  spielfeld[2];
176:            if ((summe == 3) | (summe == 30))
177:                    return summe/3;
178:
179:            // pruefe, ob unentschieden ist oder ob das Spiel noch nicht fertig i
180:            if (anzahlGespielteZuege == 9)
181:                return UNENTSCHIEDEN;
182:            else
183:                return UNKLAR;
184:        }
185:
186:        public boolean macheZug(int feld) {
```



```
187:            // Fuehrt Zug auf Feld 'feld' aus und gibt 'true' zurueck bei Erfol
188:            // Wenn Zug nicht ausgefuehrt werden kann, wird 'false' zurueckgegebe
189:
190:            if (spielfeld[feld] == LEER && status != FERTIG) {
191:                // Zug erlaubt, deshalb ausfuehre
192:                spielfeld[feld] = aktuellerSpieler;
193:                tttboard.setzeFeld(feld,aktuellerSpieler);
194:                anzahlGespielteZuege++;
195:
196:                // anschliessend kommt Gegner dra
197:                aktuellerSpieler = getGegner(aktuellerSpieler);
198:
199:                // pruefe, ob das Spiel fertig ist und wer u.U. der Sieger is
200:                int resultat;
201:                if ((resultat = istSpielFertig()) != UNKLAR)    {
202:                    String text;
203:                    status = FERTIG;
204:
205:                    switch( resultat) {
206:                        case SPIELER_X : text = new String("Spieler X hat gewonnen!"); break;
207:                        case SPIELER_O: text = new String("Spieler O hat gewonnen!"); break;
208:                        default: text = new String("Unentschieden!"); break;
209:                    }
210:                    // Meldung ueber Resultat als Dialogbox anzeige
211:                    TTTDialog sieger = new TTTDialog(this, "Spielende", true, text, "OK");
212:                    sieger.setVisible(true);
213:                    return false;
214:                }
215:
216:                // Erweiterung fuer Computer Spiele
217:                if (aktuellerSpieler == computerSpieler) {
218:                    TTTComputerPlayer cs = new TTTComputerPlayer();
219:                    macheZug(cs.besterZug(this));
220:                }
221:                return true;
222:            }
223:            return false;
224:        }
225:
226:        public void actionPerformed(ActionEvent e) {
227:            // Menu- und Button-Ereignisse behandel
228:            if ((e.getSource() == neu) | (e.getSource() == neuesSpielButton)) { neuesSpiel();
    }
229:            if (e.getSource() == laden) {
230:                TTTFileIO io = new TTTFileIO();
231:                io.spielLaden(this);
232:            }
233:            if (e.getSource() == speichern) {
234:                TTTFileIO io = new TTTFileIO();
235:                io.spielSpeichern(this);
236:            }
237:            if (e.getSource() == beenden) { System.exit(0); }
238:            if (e.getSource() == ueber) {
239:                    // Ueber-Dialog anzeige
240:                    TTTDialog ueberDialog = new TTTDialog(this, "Ueber Tic Tac Toe ...",
    true, "(c) 2000, Thomas Duebendorfer", "OK");
241:                    ueberDialog.setVisible(true);
242:            }
243:        }
244:    }
```

D:\daten\tbz\TTT\ttt_final\TTTDialogTest.java

```
1:  // TTTDialogTest.java
2:  // Klasse, um TTTDialog testen zu koenne
3:  import java.awt.*;
4:  import java.awt.event.*;
5:
6:  public class TTTDialogTest extends Frame {
7:      TTTDialogTest() {
8:          super("Dialog Test");
9:
10:         addWindowListener(new WindowAdapter() {
11:             public void windowClosing(WindowEvent e) {
12:                 System.exit(0);
13:             }
14:         });
15:     }
16:
17:     public static void main (String[] args) {
18:         TTTDialogTest dlgTest = new TTTDialogTest();
19:         dlgTest.setSize(200,200);
20:         dlgTest.show();
21:
22:         // Meldung ueber Resultat als Dialogbox anzeige
23:         TTTDialog dlg = new TTTDialog(dlgTest, "Test", true, "Dialog funktioniert!",
    "OK");
24:         dlg.setVisible(true);
25:     }
26: }
```


D:\daten\tbz\TTT\ttt_final\TTTDialog.java

```
1:  // TTTDialog.java
2:  // Klasse, die eine Dialogbox darstellt mit einem
3:  // vorgegebenen Titel, einem Meldungs-Text und einem Button mit vorgegebenem Labeltex
4:  // Der Dialog schliesst sich bei Klick auf den Button selbstaendig
5:
6:  import java.awt.*;
7:  import java.awt.event.*;
8:
9:  class TTTDialog extends Dialog implements ActionListener {
10:         // Instanzvariblen fuer ok-Button und einem Label als Meldungstex
11:         private Button ok;
12:         private Label resultat;
13:
14:         TTTDialog(Frame owner, String title, boolean isModal, String text, String
    buttonLabel) {
15:             // Der Dialog gehoert zum Frame owner (wichtig wegen Fokus
16:             // title = Fenstertitel, isModal = true bei modalem Dialo
17:             // test  = Meldungstext, buttonLabel = Beschriftung des ok-Buttor
18:
19:             //Super-Konstruktor aufrufe
20:             super(owner,title, isModal);
21:
22:             // GUI-Elemente erzeuge
23:             ok =  new Button(buttonLabel);
24:             resultat = new Label(text);
25:
26:             // GUI-Elemente zum Dialog hinzufuege
27:             add(resultat,BorderLayout.NORTH);
28:             add(ok,BorderLayout.SOUTH);
29:
30:             // Groesse des Dialoges festlege
31:             this.setSize(200,120);
32:
33:             // Ereignisbehandlung fuer ok-Button installiere
34:             ok.addActionListener(this);
35:         }
36:
37:         public void actionPerformed(ActionEvent e) {
38:             // Ereignisbehandlung: Dialog ausblenden und Ressourcen freigebe
39:             // bei Klick auf ok-Butto
40:             if (e.getSource() == ok) {
41:                 dispose();
42:             }
43:         }
44: }
45:
```

D:\daten\tbz\TTT\ttt_final\TTTBoard.java

```java
// TTTBoard.java
// Klasse, die das Spielfeld-GUI als 3x3 Buttons anleg
// und auf Button-Clicks reagiert, um Spielzuege auszufuehre

import java.awt.*;
import java.awt.event.*;

class TTTBoard extends Panel implements ActionListener {
    static char SPIELER_X  = 'X';
    static char spielerO = 'O';
    Button[] button;
    TTT ttt;
    Font myFont;

    TTTBoard(TTT ttt) {
        // Konstruktor von TTTBoar
        // Super-Konstruktor mit LayoutManager als Argument aufrufe
        super(new GridLayout(3,3));

        // Initialisierungen von Instanzvariable
        this.ttt = ttt;
        myFont = new Font("Helvetica",Font.BOLD,20);

        // Buttons fuer das Spielfeld erzeuge
        button = new Button[9];

        // Buttons initialisiere
        int i;
        for (i=0; i<9; i++) {
                // Button erzeugen und Schriftart festlege
                button[i] = new Button("");
                button[i].setFont(myFont);

                // Ereignisbehandlung installiere
                button[i].setActionCommand(i + "");
                button[i].addActionListener(this);

                // Button zu Panel hinzufuege
                add(button[i]);
        }
    }

    public void setzeFeld(int i, int spieler) {
        // Feld-Button i als besetzt durch Spieler 'spieler'
        // markieren mit "X", "O" oder "
        if (spieler == ttt.SPIELER_X ) {
            button[i].setLabel("X");
        } else if (spieler == ttt.SPIELER_O) {
            button[i].setLabel("O");
        } else if (spieler == ttt.LEER) {
            button[i].setLabel("");
        }
    }

    public void actionPerformed(ActionEvent e) {
        // Ereignisbehandlung fuer Klicks auf Button
        // Fuehre einen Zug aus mit ttt.macheZug(i), wenn Feld-Button i geklickt wurde
        if (e.getSource() instanceof Button) {
            ttt.macheZug(Integer.parseInt(e.getActionCommand()));
        }
    }
}
```


D:\daten\tbz\TTT\ttt_final\TTTComputerPlayer.java

```java
// TTTComputerPlayer.java
// Einfacher Computerspieler zu Tic Tac Toe mit minimaler Intelligen
// Computer gewinnt immer, wenn er dazu nur noch einen Zug machen mus
// Wenn der Gegner noch einen Zug zum Siegen braucht, dann macht d
// Computer diesen Zug, um den Gegner zu blockieren
// Ansonsten waehlt er zufaellig einen Zug aus

public class TTTComputerPlayer {
    TTTComputerPlayer() {
    }

    public int siegInEinemZug(TTT ttt, int aktuellerSpieler) {
        // gibt Zug (Feld) zum Gewinn zurueck oder -1 falls kein Sieg in 1 Zug moegli
        for (int i =0; i <9; i++) {
            if(ttt.spielfeld[i] == TTT.LEER) {
                // Feld ist leer, mache eine Zug darauf und teste, ob gewonne
                ttt.spielfeld[i] = aktuellerSpieler;
                if( ttt.istSpielFertig() == aktuellerSpieler) {
                    ttt.spielfeld[i] = TTT.LEER;
                    return i;
                }
                ttt.spielfeld[i] = TTT.LEER;
            }
        }
        return -1;
    }

    public int besterZug(TTT ttt) {
        int feld;
        // Teste, ob Computer in 1 Zug gewinnen kan
        if((feld = siegInEinemZug(ttt, ttt.getAktuellerSpieler())) != -1)
            return feld;

        // Teste, ob Gegner in 1 Zug gewinnen kann (und ziehe ggf. dorthi
        if((feld = siegInEinemZug(ttt, ttt.getGegner(ttt.getAktuellerSpieler()))) != -1)
            return feld;

        // Computer kann in 1 Zug nicht gewinne
        // mach einen Zufallszu
        do  {
            feld = (int)(Math.random()*9); // 0.0 <= Math.random() < 1
        }  while(ttt.spielfeld[feld] != TTT.LEER);

        return feld;
    }
}
```

```
 1:  // TTTFileIO.java
 2:  // Klasse zum Laden und Speichern von TTT-Spielstaende
 3:  import java.awt.*; // FileDialog
 4:  import java.io.*;  // FileInputStream, ...
 5:
 6:  public class TTTFileIO {
 7:      TTTFileIO() {
 8:      }
 9:
10:      public static void spielLaden(TTT ttt) {
11:          // Spiel laden
12:          File f;
13:          // File Dialog zum Laden erzeugen und anzeige
14:          FileDialog fd = new FileDialog(ttt, "Spiel laden ...",FileDialog.LOAD);
15:          fd.show();
16:
17:          try {
18:              // FileInputStream, BufferedInputStream zur Datei f.getPath() erzeug
19:              if (fd.getFile() == null) return; // Keine Datei ausgewählt
20:              else f = new File(fd.getDirectory(), fd.getFile());
21:              FileInputStream iStream = new FileInputStream(f.getPath());
22:              BufferedInputStream iBufStream = new BufferedInputStream(iStream);
23:
24:              // Spielstand in String spielStand einlese
25:              String spielStand = new String("");
26:              byte teil[] = new byte[256];
27:              int anzGeleseneBytes = 0;
28:              while(iBufStream.available() > 0) {
29:                  anzGeleseneBytes = iBufStream.read(teil);
30:                  spielStand += new String(teil,0,anzGeleseneBytes);
31:              }
32:
33:              // Spielfeld entsprechend geladenem spielStand vorbereite
34:              // Initialisieren als neues Spie
35:              ttt.neuesSpiel();
36:
37:              // Felder mit ttt.tttboard.setzeFeld(...) auf "X", "O" oder "" setz
38:              int anzX = 0;
39:              int anzO = 0;
40:              for (int feld=0; feld<9; feld++) {
41:                  if (spielStand.charAt(feld) == 'X') {
42:                      ttt.spielfeld[feld] = TTT.SPIELER_X;
43:                      ttt.tttboard.setzeFeld(feld, TTT.SPIELER_X);
44:                      anzX++;
45:                  } else if (spielStand.charAt(feld) == 'O') {
46:                      ttt.spielfeld[feld] = TTT.SPIELER_O;
47:                      ttt.tttboard.setzeFeld(feld, TTT.SPIELER_O);
48:                      anzO++;
49:                  } else {
50:                      ttt.spielfeld[feld] = TTT.LEER;
51:                      ttt.tttboard.setzeFeld(feld, TTT.LEER);
52:                  }
53:              }
54:
55:              // ttt.anzahlGespielteZuege aktualisiere
56:              ttt.anzahlGespielteZuege = anzX + anzO;
57:
58:              // ttt.aktuellerSpieler aktualisiere
59:              if (anzO > anzX) ttt.aktuellerSpieler = TTT.SPIELER_X;
60:              else if (anzO < anzX) ttt.aktuellerSpieler = TTT.SPIELER_O;
61:
62:              // testen, ob Spiel schon fertig ist und in diesem Fal
63:              // verhindern, dass ein fertiges Spiel weitergespielt werden ka
```



```
64:              if (ttt.istSpielFertig() != TTT.UNKLAR) {
65:                  ttt.status = ttt.FERTIG;
66:              }
67:          } catch (IOException ioEvent) {
68:              // Fehler-Meldung als Dialogbox anzeige
69:              TTTDialog dateiFehler = new TTTDialog(ttt, "Dateifehler", true,
     ioEvent.toString(), "OK");
70:              dateiFehler.setSize(400,120);
71:              dateiFehler.show();
72:          }
73:      }
74:
75:      public void spielSpeichern(TTT ttt) {
76:          // Spiel speicherr
77:          File f;
78:          // File Dialog zum Speichern erzeugen und anzeige
79:          FileDialog fd = new FileDialog(ttt, "Spiel speichern ...",FileDialog.SAVE);
80:          fd.show();
81:
82:          if (fd.getFile() == null) return; // Keine Datei ausgewählt
83:          else f = new File(fd.getDirectory(), fd.getFile());
84:          // Spielstand der Felder 0 bis 8 im String spielStand codiere
85:          // Bsp. spielStand=="XXOOXX O ", wenn ein X oben links steht und in unterst
86:          // Zeile noch links und rechts aussen ein freies Feld is
87:          String spielStand = new String("");
88:          for (int feld=0; feld<9; feld++) {
89:              if (ttt.spielfeld[feld] == TTT.SPIELER_X) {
90:                  spielStand += new String("X");
91:              } else if (ttt.spielfeld[feld] == TTT.SPIELER_O) {
92:                  spielStand += new String("O");
93:              } else {
94:                  spielStand += new String(" ");
95:              }
96:          }
97:
98:          try {
99:              // FileOutputStream, BufferedOutputStream zur Datei f.getPath() erzeug
100:             FileOutputStream oStream = new FileOutputStream(f.getPath());
101:             BufferedOutputStream oBufStream = new BufferedOutputStream(oStream);
102:
103:             // String spielStand speichern und File schliesse
104:             oBufStream.write(spielStand.getBytes());
105:             oBufStream.flush();
106:             oBufStream.close();
107:         } catch (IOException ioEvent) {
108:             // Fehler-Meldung als Dialogbox anzeige
109:             TTTDialog dateiFehler = new TTTDialog(ttt, "Dateifehler", true,
     ioEvent.toString(), "OK");
110:             dateiFehler.setSize(400,120);
111:             dateiFehler.show();
112:         }
113:     }
114: }
115:
116:
```