```
/* TTT.java                                  *
/* Hauptklasse fuer Tic Tac Toe mit Java AWT (GUI) *
/* 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

        // Menu definierer
        // MenuBar erzeugen, Menus und MenuItems erzeuge

        // Ereignisbehandlung hinzufuegen zu MenuItem

        // Menupunkte zu Menus hinzufuegen

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

        // DropDown-Liste, Neues-Spiel-Button definiere

        // Ereignisbehandlung fuer NeuesSpielButto

        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.show();
    }

    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
            return SPIELER_X;
    }

    public void neuesSpiel() {
        // Initialisierungen, um ein neues Spiel beginnen zu koenne
        computerSpieler = 0; // kein ComputerSpieler als Standar
        anzahlGespielteZuege = 0;
        loescheSpielfeld();
        // Startspieler aus DropDown auslese
        // folgende Kommentar koennen entfernt werden, sobald spielerChoice implementie
wurde:
        //int aktuellerSpielerIndex = spielerChoice.getSelectedIndex(
        //if (aktuellerSpielerIndex == 0)     { aktuellerSpieler = SPIELER_X;
        //else if (aktuellerSpielerIndex == 1) { aktuellerSpieler = SPIELER_O;
        //else if (aktuellerSpielerIndex == 2) { aktuellerSpieler = SPIELER_
computerSpieler = SPIELER_O;
        //else if (aktuellerSpielerIndex == 3) { aktuellerSpieler = SPIELER_
computerSpieler = SPIELER_X;
        // solange spielerChoice noch nicht implementiert ist
        aktuellerSpieler = SPIELER_X;

        status = BEREIT;
    }

    public void loescheSpielfeld() {
        // ganzes Spielfeld (inkl. GUI-Board) loesche
        for (int i = 0; i < 9; i++) {
            spielfeld[i] = 0;
            tttboard.setzeFeld(i,LEER);
        }
    }

    public int istSpielFertig() {
        // gibt Gewinner SPIELER_X  bzw. SPIELER_O zuruec
        // oder UNENTSCHIEDEN, falls UNENTSCHIEDEN is
        // oder UNKLAR, falls Spiel noch nicht FERTIG is

        // pruefe, ob es 3 gleiche Symbole in einer Zeile ha

        // pruefe, ob es 3 gleiche Symbole in einer Spalte hat

```

```
123:          // pruefe, ob es 3 gleiche Symbole in einer Diagonale hat
124:
125:          // pruefe, ob unentschieden ist oder ob das Spiel noch nicht fertig i
126:
127:          return UNKLAR;
128:      }
129:
130:      public boolean macheZug(int feld) {
131:          // Fuehrt Zug auf Feld 'feld' aus und gibt 'true' zurueck bei Erfol
132:          // Wenn Zug nicht ausgefuehrt werden kann, wird 'false' zurueckgegebe
133:
134:          if (spielfeld[feld] == LEER && status != FERTIG) {
135:              // Zug erlaubt, deshalb ausfuehre
136:              spielfeld[feld] = aktuellerSpieler;
137:              tttboard.setzeFeld(feld,aktuellerSpieler);
138:              anzahlGespielteZuege++;
139:
140:              // anschliessend kommt Gegner dra
141:              aktuellerSpieler = getGegner(aktuellerSpieler);
142:
143:              // pruefe, ob das Spiel fertig ist und wer u.U. der Sieger i
144:              int resultat;
145:              if ((resultat = istSpielFertig()) != UNKLAR)    {
146:                  String text;
147:                  status = FERTIG;
148:
149:                  switch( resultat) {
150:                      case SPIELER_X : text = new String("Spieler X hat gewonnen!"); break;
151:                      case SPIELER_O: text = new String("Spieler O hat gewonnen!"); break;
152:                      default: text = new String("Unentschieden!"); break;
153:                  }
154:                  // Meldung ueber Resultat als Dialogbox anzeige
155:                  TTTDialog sieger = new TTTDialog(this, "Spielende", true, text, "OK");
156:                  sieger.show();
157:                  return false;
158:              }
159:
160:              // Erweiterung fuer Computer Spiele
161:              if (aktuellerSpieler == computerSpieler) {
162:                  TTTComputerPlayer cs = new TTTComputerPlayer();
163:                  macheZug(cs.besterZug(this));
164:              }
165:              return true;
166:          }
167:          return false;
168:      }
169:
170:      public void actionPerformed(ActionEvent e) {
171:          // Menu- und Button-Ereignisse behandel
172:          // ...
173:          if (e.getSource() == laden) {
174:              //TTTFileIO io = new TTTFileIO()
175:              //io.spielLaden(this),
176:          }
177:          if (e.getSource() == speichern) {
178:              //TTTFileIO io = new TTTFileIO()
179:              //io.spielSpeichern(this);
180:          }
181:          // ...
182:      }
183:  }
```

D:\daten\tbz\TTT\ttt_framework\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_framework\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-Butto
18:
19:              //Super-Konstruktor aufrufe
20:              super(owner,title, isModal);
21:
22:              // GUI-Elemente erzeuge
23:
24:              // GUI-Elemente zum Dialog hinzufuege
25:
26:              // Groesse des Dialoges festlege
27:
28:              // Ereignisbehandlung fuer ok-Button installiere
29:
30:          }
31:
32:          public void actionPerformed(ActionEvent e) {
33:              // Ereignisbehandlung: Dialog ausblenden und Ressourcen freigebe
34:              // bei Klick auf ok-Butto
35:              if (e.getSource() == ok) {
36:                  dispose();
37:              }
38:          }
39:  }
40:
```

D:\daten\tbz\TTT\ttt_framework\TTTBoard.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
        // ...
    }
}
```


D:\daten\tbz\TTT\ttt_framework\TTTComputerPlayer.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 speicher
77:          // File Dialog zum Speichern erzeugen und anzeige
78:
79:          // Spielstand der Felder 0 bis 8 im String spielStand codiere
80:          // Bsp. spielStand=="XXOOXX O ", wenn ein X oben links steht und in unterst
81:          // Zeile noch links und rechts aussen ein freies Feld is
82:          String spielStand = new String("");
83:          // ...
84:
85:          // Das folgende try-catch wird erst gebraucht, wenn die
86:          // abgefangenen Exceptions auftreten koennen
87:          // try {
88:              // FileOutputStream, BufferedOutputStream zur Datei f.getPath() erzeug
89:
90:              // String spielStand speichern und File schliesse
91:
92:
93:          // } catch (IOException ioEvent)
94:              // Fehler-Meldung als Dialogbox anzeige
95:          //  TTTDialog dateiFehler = new TTTDialog(ttt, "Dateifehler", true
    ioEvent.toString(), "OK"),
96:          //  dateiFehler.setSize(400,120)
97:          //  dateiFehler.show(),
98:          // }
99:      }
100: }
101:
102:
```