```
// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                 --Klasse "Action"--                                        *
* Die eigentliche "Hauptklasse" in der der Spielablauf gesteuert wird. Sie    *
* enthaelt alle Methoden, die notwendig sind, um das Zeichnen zu ermöglichen. *
* Da eine Mehrfachvererbung in Java nur ueber "Umwege" (Instanzenbildung oder *
* Interfaces) möglich ist und dies einen unnoetigen und kompliezierten und    *
* zudem umfangreicheren Code erfordern würde, sind hier bewusst alle Methoden *
* in einer Klasse vereinigt. Lediglich die Methode "punkte" wurde als Beispiel*
* in einer eigenen Klasse (SpielPunkte) realiesiert und wird dann ueber Instan-*
* zenbildung aufgerufen.                                                     *
* Fast alle notwendigen Variablen erbt "Action" von "Welt". Lediglich die     *
* Punktestände werden in "Action" als "static" deklariert, da diese als       *
* Klassenvariablen (und nicht als Instanzvariablen wie alle Anderen)von       *
* "Spielpunkte" geerbt werden müssen.                                        *
*                                                                            *
*****************************************************************************/
import java.awt.*;
import java.awt.event.*;
import java.util.*;                                                  //für Zufallszahlengenerator

class Action extends Welt implements MouseListener,MouseMotionListener,Runnable
{
 static int strafpunkte,ufo,stern,komet,enterprise,lebenNeu;         //Klassenvariablen für Spielstand
 Random zufall = new Random();                                       //neue Instanz
 Speed speed = new Speed();                                          //neue Instanz
 Sound sound = new Sound();                                          //neue Instanz
 Frame rahmen = new NamenFrame();                                    //neue Instanz



/*****************************************************************************
*                -Konstruktor-                                               *
*****************************************************************************/
 public Action()//*************************************************************
  {
   addMouseListener(this);                                           //Mouselistener für Mausklicks
   addMouseMotionListener(this);                                     //Mouselistener für Mausbewegung
  }



/*****************************************************************************
*           -Methode "paint"-                                                *
* Methode die das zeichnen übernimmt.                                        *
```

```
*****************************************************************************/
 public void paint(Graphics g) //*********************************************
  {
   if(esPoint==0)rahmen.show();                                          //Fenster Namen bei erststart
   Graphics ge = getGraphics();
   if(nzStart==1){t.stop();u=0;}                                         //Neuzeichnen bei Start
   if(neuzeichnen==1){t.stop();}                                         //Neuzeichnen bei Spiel
   neuzeichnen=1;
   nsPoint=0;                                                            //Maus wird freigegeben
   ge.drawImage(img,0,0,this);                                           //zeichne Hintergrund
   trefferAnzeige();                                                     //Methodenaufruf
   lebenAnzeige();
   geschwindigkeitAnzeige();
   zeitAnzeige();
   schussAnzeige();
   start();                                                              //ruft "run" auf (Thread)
  }



/*****************************************************************************
*              -Methode "run"-                                              *
* Methode, die von "start" aufgerufen wird. Sie arbeitet als Thread, um die    *
* Gleichzeitigkeit mehrerer Prozesse zu ermöglichen. Hier werden der Startpunkt*
* der Objekte als Random ermittelt, sowie die Bewegungungen gesteuert.         *
*****************************************************************************/
 public void run() //*************************************************************
  {
   Graphics ge = getGraphics();
   erstStart();                                                          //Methodenaufruf
   esPoint=1;                                                            //Erststart wird blockiert
   ge.drawImage(img9,0,60,this);                                         //aktualisiert Sternenhimmel
   if(nzPause==1){ge.drawImage(img23,0,0,this);}                         //Neuzeichnen bei Pause
   if(nzGameOver==1){gameOver();}                                        //Neuzeichnen bei Game Over
   ufo();
   a =(zufall.nextDouble()*800);                                         //ermittelt die AnfangsKoordinaten
   x = (int)a-300;                                                       //der Objekte als Random
   b =(zufall.nextDouble()*800);
   y = (int)b-300;
   if ((x>-200 && x<0) &&  (y>100 && y<450))                             //Einschraenkung des Bereichs
          {                                                              //Zuweisung der Objekte in
           if (x>-200 && x<=-150) {feind = img1;}                        //Abhaengigkeit von den
           if (x>-150 && x<=-100) {feind = img2;}                        //Koordinaten
           if (x>-100 && x<=-50) {feind = img21;}
           if (x>-50 && x<0) {feind = img20;}
           for (c=0 ;c<laenge+201  ;c++  )                               //solange Objekt nicht am
           {                                                             //Spielfeldende
            if (time>-1)                                                 //wenn noch Zeit
            {cursorHand();                                               //Methodenaufruf
             x++;                                                        //Objekt ein Schritt weiter
```

```
              schuss();
              feind();
              kreuzNeu();
              aufprall();
              geschwindigkeitAnzeige();
              zeitAnzeige();
              speed.run(geschwindigkeit);
              if ( x==laenge ) {schuss=0;run();};                    //wenn Objekt am Ende-Neuaufruf der Methode
             }
             if (leben==0){time = -1;}
             if (time==-1){gameOver();}                              //wenn Anzahl Leben = 0-Zeit abgelaufen
            }                                                        //wenn Zeit abgelaufen- Spielende
           }
   if (time>-1) run();                                               //wenn noch Zeit-Neuaufruf
  }



/*****************************************************************************
*                    -1. Methode "feind"-                                    *
* Methode, die das "Feind"-Objekt zeichnet                                   *
*****************************************************************************/
  public void feind () //*************************************************
  {
   if(pPoint==0)                                                     //wenn nicht Pause
   {
    Graphics ge = getGraphics();                                     //zeichne Feind
    ge.drawImage(feind,x,y,this);
    if((feind==img1) || (feind==img21))                              //wenn Ufo oder Enterprise
     {
      if(x>1)
       {                                                             //zeichne Feindschuss
        ge.drawImage(img27,xSchussAlt,y+5,this);
        ge.drawImage(img26,x+schuss+85,y+5,this);
        xSchussAlt=x+schuss+75;
        schuss=schuss+3;
        abschuss();
       };
     }
   }
  }



/*****************************************************************************
*                    -2. Methode "feind"-                                    *
* Methode, die das "Feind"-Objekt zeichnet                                   *
*****************************************************************************/
/*public void feind ()  //*************************************************
  {
```

```
   if(pPoint==0)                                                    //wenn nicht Pause
   {
    Graphics ge = getGraphics();                                    //zeichne Feind
    ge.drawImage(feind,x,y,this);                                   //wenn Ufo oder Enterprise
    if((feind==img1) || (feind==img21))
     {
      if(x>1)
       {
        ge.setColor(Color.black);                                   //zeichne Feindschuss
        ge.fillOval(xSchussAlt,y+5,10,10);
        ge.setColor(new Color(255,165,0));
        ge.fillOval(x+schuss+85,y+5,10,10);
        xSchussAlt=x+schuss+75;
        schuss=schuss+1;
        abschuss();
       };
     }
   }
  } */




/*************************************************************************
*             -Methode "schuss"-                                         *
* Methode, die kontrolliert wann geschossen wurde,wieviel Munition noch vorhan-*
* den ist, ob ein Feind getroffen wurde und eine Explosions-Animation auslöst  *
*************************************************************************/
 public void schuss() //*************************************************
  {
   Graphics ge = getGraphics();
   if (munition<1 ){sPoint=1;                                       //wenn keine Munition vorhanden
                    speed.run(500);
                    ge.drawImage(img19,0,60,this);
                    speed.run(2000);
                    munition=20;                                    //neue Munition
                    strafpunkte=strafpunkte+5;                      //5 Strafpunkte
                    ge.drawImage(img9,0,60,this);
                    ge.setFont(f1);                                 //aktualisiere Schussanzeige
                    ge.setColor(Color.black);
                    ge.drawString(" Schuss:   "+0,546,37);
                    ge.setColor(new Color(220,20,60));
                    ge.drawString(" Schuss:   "+munition,546,37);
                    }
   yPlus = yCo+30;
   yMinus = yCo-50;
   xPlus = xCo+30;
   xMinus = xCo-50;
   if ((click==1)&&(yPlus>y && y>yMinus)&&(xPlus>x && x>xMinus))     //wenn Mausclick auf Feindobjekt
       {
```

```
        if(feind==img1){ufo++;}
        if(feind==img2){komet++;}                                   //zaehle den Treffer
        if(feind==img20){stern++;}
        if(feind==img21){enterprise++;}
        feind = img5;                                               //lösche Feind
        ge.drawImage(feind,x,y,this);                               //Explosions-Sound
        sound.explosion();
        crash();                                                    //Explosions-Animation starten
        schuss=0;
        if (time>-1) run();
       }                                                            //wenn noch Zeit-Neustart
   click=0;
   sPoint=0;                                                        //Pointer auf 0 (siehe mousePressed())
  }



/*****************************************************************************
*                      -Methode "aufprall"-                                  *
* Methode, die kontrolliert, ob es einen Zusammenstoss gab.                  *
*****************************************************************************/
 public void aufprall() //*********************************************
  {
   Graphics ge = getGraphics();
   xCrash=x+60;
   yCrash=y;
   tre=0;
   if(x>740 && (yCo+30>y) && (y>yCo-50))                            //wenn Zusammenstoss
    {
     sPoint=1;
     leben--; lebenNeu=leben;                                       //ein Leben weniger
     l=1;
     sound.explosion();                                             //Explosionssound
     explosion ();                                                  //Explosionsanimation
     sound.getroffen();
     tre=1;
     ge.drawImage(img22,0,60,this);
     ge.setColor(Color.red);
     ge.setFont(f2);
     ge.drawString("Anzahl Leben :   "+leben,280,400);              //Aktualisiere Lebensanzeige
     speed.run(2000);
     sPoint=0;
     ge.drawImage(img9,0,60,this);                                  //lösche Bildschirm
     lebenAnzeige();
     schuss=0;
    }
   if (time>-1 && tre==1) run();                                    //wenn noch Zeit-Neustart
  }
```

```
/*****************************************************************************
*               -Methode "abschuss"-                                         *
* Methode, die kontrolliert, ob es einen Abschuss gab(analog zu Aufprall).    *
*****************************************************************************/
 public void abschuss() //*************************************************
  {
   Graphics ge = getGraphics();
   xCrash=xSchussAlt+30;
   yCrash=y;
   tre=0;
   if(((xSchussAlt-60)>740)&&((xSchussAlt-60)<840)&&(yCo+30>y) && (y>yCo-50))
    {
     sPoint=1;
     leben--; lebenNeu=leben;
     l=1;
     sound.explosion();
     explosion ();
     sound.getroffen();
     tre=1;
     ge.drawImage(img22,0,60,this);
     ge.setColor(Color.red);
     ge.setFont(f2);
     ge.drawString("Anzahl Leben :   "+leben,280,400);
     speed.run(2000);
     sPoint=0;
     ge.drawImage(img9,0,60,this);
     lebenAnzeige();
     schuss=0;
    }
   if (time>-1 && tre==1) run();
  }



/*****************************************************************************
*                 -Methode "gameOver"-                                       *
* Methode,die aufgerufen wird, wenn das Spiel beendet ist.                   *
*****************************************************************************/
 public void gameOver()//*************************************************
 {
  nzGameOver=1;                                                  //ermöglicht Neuzeichnen
  setCursor(new Cursor(Cursor.DEFAULT_CURSOR));                  //setze Standartcursor
  Graphics ge = getGraphics();
  speed.run(2000);                                               //warte
  ge.drawImage(img3,0,0,this);
  SpielPunkte sp = new SpielPunkte();                            //neue Instanz der Klasse SpielPunkte
  sp.punkte(ge);                                                 //Aufruf der Methode punkte von Spielpunkte
  Highscore hs = new Highscore();                                //neue Instanz der Klasse Highscore
  hs.lesen();                                                    //Datei lesen
```

```
  ge.setColor(new Color(255,204,0));
  ge.setFont(f3);
  if(Highscore.w<SpielPunkte.total)                                    //wenn neuer Punktestand>Highscore
     {
      highscoreAnim();                                                 //Highscore Animation
      hs.schreiben();                                                  //schreibe Punktestand in Highscore
      ge.drawString(""+(SpielPunkte.total)+" Punkte von",305,90);      //zeige neuen Highscore
     }
  else {ge.drawString(""+(Highscore.w)+" Punkte von",305,90);}         //sonst alter Highscore
  NamenPanel np = new NamenPanel();                                    //neue Instanz der Klasse NamenPanel
  if(Highscore.w<SpielPunkte.total)                                    //wenn neuer Punktestand>Highscore
    {
     np.schreiben();                                                   //schreibe Namen in Datei
     ge.drawString(""+(NamenPanel.na),460,90);                         //zeige Namen
    }
  else{                                                                //sonst
       np.lesen();                                                     //lese Name aus Datei
       ge.drawString(""+(NamenPanel.na1),460,90);                      //zeige Name
      }
  sound.gameOver();                                                    //Game over-Sound
  t.suspend();                                                         //System-Pause
 }



/*************************************************************************
*                  -Methode "zeitAnzeige"-                               *
* Methode, die die Zeitanzeige zeichnet                                  *
*************************************************************************/
 public void zeitAnzeige()//*****************************************************
  {
   if(pPoint==0)                                                       //wenn nicht Pause
    {
     Graphics ge = getGraphics();
     ge.setFont(f1);                                                   //erhöhe Zeitindex
     e++;
     ge.setColor(new Color(220,20,60));                                //aktualisiere Zeitausgabe
     ge.drawString("  Zeit:   "+time,21,37);
     if (e==150){                                                      //wenn Zeitindex auf 150
                 e=0;time--;                                           //setze Zeitindex auf 0 und ziehe 1 Zeitpunkt ab
                 if((time<100)&&((time%5)==0))                         //wenn Zeit<100 und Mod%5 wahr
                              {
                               sound.uhr();                            //Uhrensound
                              }
                 if (time==-1){gameOver();}                            //wenn Zeit abgelaufen-Game over
                 if (time>-1)                                          // Zaehler wird neu gezeichnet solange Spiel läuft
                  {
                   ge.setColor(Color.black);
                   ge.drawString("  Zeit:   "+(time+1),21,37);
                   ge.setColor(new Color(220,20,60));
```

```
                    ge.drawString("  Zeit:   "+time,21,37);

                   }
                 }
     }
   }



/*****************************************************************************
*                   -Methode "geschwindigkeitAnzeige"-                       *
* Methode, die die Geschwindigkeitsanzeige zeichnet                          *
*****************************************************************************/
 public void geschwindigkeitAnzeige() //*****************************************
  {
   if(pPoint==0)                                                    //wenn nicht Pause
    {
     s++;                                                           //Geschwindigkeitsindex erhöhen
     Graphics ge = getGraphics();
     ge.setFont(f1);
     ge.setColor(new Color(50,205,50));                             //aktualisiere Anzeige
     ge.drawString("  Speed:  "+speedAnzeige,155,37);
     if (s==1500 && geschwindigkeit>1 )                             //wenn Index auf 1500 und noch nicht höchste Geschwindigkeit
       {
        s=0;speedAnzeige++;                                         //ein Schritt schneller
        ge.setColor(Color.black);
        ge.drawString("  Speed:  "+(speedAnzeige-1),155,37);        //aktualisiere Anzeige
        ge.setColor(new Color(50,205,50));
        ge.drawString("  Speed:  "+speedAnzeige,155,37);
        geschwindigkeit--;
       }
    }
  }



/*****************************************************************************
*                   -Methode "lebenAnzeige"-                                 *
* Methode, die die Lebensanzeige zeichnet                                    *
*****************************************************************************/
  public void lebenAnzeige() //***************************************************
  {
   Graphics ge = getGraphics();
   ge.setFont(f1);
   ge.setColor(new Color(50,205,50));
   ge.drawString("  Leben:   "+leben,417,37);
   if (leben>-1 && l== 1){                                          //wenn noch Lebeb vorhanden
                 ge.setColor(Color.black);                          //aktualisiere Anzeige
                 ge.drawString("  Leben:   "+(leben +1),417,37);
                 ge.setColor(new Color(50,205,50));
```

```
                ge.drawString("  Leben:   "+leben,417,37);
                l=0;
                }
  }



/*****************************************************************************
*                  -Methode "schussAnzeige"-                                 *
* Methode, die die Munitionsanzeige zeichnet                                 *
*****************************************************************************/
  public void schussAnzeige() //************************************************
  {
   Graphics ge = getGraphics();
   ge.setFont(f1);
   ge.setColor(new Color(220,20,60));
   ge.drawString(" Schuss:   "+munition,546,37);
   if (munition>=0){                                                  //wenn Munition vorhanden
                ge.setColor(Color.black);                             //aktualisiere Anzeige
                ge.drawString(" Schuss:   "+(munition +1),546,37);
                ge.setColor(new Color(220,20,60));
                ge.drawString(" Schuss:   "+munition,546,37);
                l=0;
                }
  }



/*****************************************************************************
*                  -Methode "trefferAnzeige"-                                *
* Methode, die die Trefferanzeige zeichnet                                   *
*****************************************************************************/
  public void trefferAnzeige() //************************************************
  {
   Graphics ge = getGraphics();
   ge.setFont(f1);
   ge.setColor(Color.black);
   ge.drawString(" Treffer:   "+(treffer-1),286,37);
   ge.setColor(new Color(50,205,50));
   ge.drawString(" Treffer:   "+treffer,286,37);
  }



/*****************************************************************************
*                  -Methode "ufo"-                                           *
* Methode, die das Ufo zeichnet                                              *
*****************************************************************************/
 public void ufo() //***********************************************************
  {if (munition>0 && pPoint==0 && sPoint==0)                         //wenn noch Munition und nicht Pause oder Neustart
```

```
   {
    Graphics ge = getGraphics();                                        //aktualisiere Ufo an seinen Koordinaten
    ge.drawImage(img5,800,clear,this);
    ge.drawImage(img4,800,(yCo-25),this);
    clear=(yCo-25);
   }
  }



/*****************************************************************************
*                  -1. Methode "kreuzNeu"-                                   *
* Methode, die das Zielkreuz zeichnet                                        *
*****************************************************************************/
  public void kreuzNeu() //*****************************************************
  {

  if((pPoint==0)&& (yCo>100 && yCo<515 && yKreuzAlt<515))                     //wenn nicht Pause und Maus auf Spielfeld
   {
     Graphics ge = getGraphics();
     if ( yCo>90 && yCo<525  ){                                             //zeichne Kreuz an Mauskoordinaten
                              ge.drawImage(img7,clear1,clear2,this);
                              ge.drawImage(img6,(xCo-30),(yCo-30),this);
                              clear2=(yCo-30);
                              clear1=(xCo-30);
                              }
     else {ge.drawImage(img7,clear1,clear2,this);}                          //sonst lösche Kreuz
    }
  }



/*****************************************************************************
*                  -2. Methode "kreuzNeu"-                                   *
* Methode, die das Zielkreuz zeichnet                                        *
*****************************************************************************/
/*public void kreuzNeu() //*****************************************************
  {
   if((pPoint==0)&& (yCo>100 && yCo<515 && yKreuzAlt<515))
   {
    Graphics ge = getGraphics();
    ge.setColor(Color.black);
    ge.drawRect(xKreuzAlt-20,yKreuzAlt-20,40,40);
    ge.drawLine(xKreuzAlt,yKreuzAlt+30,xKreuzAlt,yKreuzAlt-30);
    ge.drawLine(xKreuzAlt+30,yKreuzAlt,xKreuzAlt-30,yKreuzAlt);
    ge.drawOval(xKreuzAlt-9,yKreuzAlt-9,18,18);
    ge.setColor(new Color(34,139,34));
    ge.drawRect(xKreuzNeu-20,yKreuzNeu-20,40,40);
    ge.drawLine(xKreuzNeu,yKreuzNeu+30,xKreuzNeu,yKreuzNeu-30);
    ge.drawLine(xKreuzNeu+30,yKreuzNeu,xKreuzNeu-30,yKreuzNeu);
```

```
    ge.drawOval(xKreuzNeu-9,yKreuzNeu-9,18,18);
    clear2=(yCo-30);
    clear1=(xCo-30);
    }
  } */



/*****************************************************************************
*                   -Methode "feuer"-                                        *
* Methode, die den Laser des Ufo zeichnet                                    *
*****************************************************************************/
 public void feuer() //*************************************************
  {
   Graphics ge = getGraphics();
   if ( yCo>100 && yCo<515 && munition>=0 && sPoint==0)                        //wenn Maus im Spielfeld und noch Munition und nicht Neustart
        {
         schussAnzeige();                                                    //zeichne Laser
         ge.setColor(Color.green);
         ge.drawLine(826,(yCo-15),clear1+40,(clear2+32));
         speed.run(30);
         ge.setColor(Color.red);
         ge.drawLine(826,(yCo-15),clear1+40,(clear2+32));
         speed.run(30);
         ge.setColor(Color.yellow);
         ge.drawLine(826,(yCo-15),clear1+40,(clear2+32));
         speed.run(30);
         ge.setColor(Color.black);
         ge.drawLine(826,(yCo-15),clear1+40,(clear2+32));
        }
  }



/* ***************************************************************************
*               -Methode "mousePressed"-                                     *
* Methode, die bei Drücken der Maus aufgerufen wird                          *
*****************************************************************************/
 public void mousePressed( MouseEvent e) //*************************************
 {
  if(nsPoint==0&& pPoint==0 && sPoint==0)                                     //wenn Maus freigegeben
   {
    if(yCo>100 && yCo<515 && yKreuzAlt<515)
       { munition--; sound.start();}                                         //ein Schuss weniger
    xExit = e.getX();
    yExit = e.getY();                                                        //wenn noch Zeit
    if(time>-1){
                feuer();                                                     //Laseranimation
                click=1;                                                     //Pointer auf 1,solange noch gespielt wird
                Graphics ge = getGraphics();
```

```
                if(munition>0)                                          //solange noch Munition vorhanden
                              {
                               ge.drawImage(img9,0,60,this);            //Hintergrund neu
                               ufo();                                   //Ufo neu
                              };
              }
   }
 }



/*************************************************************************
*          -Methode "mouseClicked"-                                      *
* Methode, die aufgerufen wird, wenn Maus gedrückt und wieder losgelassen wird *
*************************************************************************/
 public void mouseClicked(MouseEvent e) //*************************************
 {

  xExit = e.getX();
  yExit = e.getY();
  ende();                                                               //Button Exit
  neustart();                                                           //Button Start
  pause();                                                              //Button Pause
 }



/*************************************************************************
* Die folgenden Methoden weden nicht benötigt, müssen aber initzialisiert,     *
* werden,da beide Mouse-Listener von Static-Kontext sind und somit alle        *
* Methoden benötigen                                                    *
*************************************************************************/
 public void mouseEntered(MouseEvent e){}
 public void mouseExited(MouseEvent e){}
 public void mouseReleased(MouseEvent e){}
 public void mouseDragged(MouseEvent e){}



/*************************************************************************
*               -Methode "mouseMoved"-                                  *
* Methode, die bei Bewegung der Maus aufgerufen wird                    *
*************************************************************************/
 public void mouseMoved(MouseEvent e)  //*************************************
  {
   if(time>-1){                                                         //wenn noch Zeit
               xKreuzAlt=xKreuzNeu;                                     //fuer kreuzNeu()
               yKreuzAlt=yKreuzNeu;                                     //fuer kreuzNeu()
               xKreuzNeu=e.getX();                                      //fuer kreuzNeu()
               yKreuzNeu=e.getY();                                      //fuer kreuzNeu()
```

```
              xCo = e.getX();
              yCo = e.getY();
              ufo();                                                    //zeichne ufo
              if(sPoint==0)                                             //wenn nicht Neustart
              {
                kreuzNeu();                                             //zeichne Kreuz
              }
             }
  }



/*****************************************************************************
*            -Methode "start"-                                               *
* Methode, die "run" als Thread aufruft um bei allen Bewegungen auf dem      *
* Spielfeld gleichzeitig noch andere Prozesse ausführen zu können            *
*****************************************************************************/
 public void start() //*************************************************************
  {
   t = new Thread(this);
   t.start();
  }



/*****************************************************************************
*            -Methode "crash"-                                               *
* Methode die die Explosions-Animation auslöst und die Trefferanzeige        *
* aktualisiert                                                               *
*****************************************************************************/
 public void crash() //*************************************************************
  {
   xCrash=xCo-20;                                                           //übergibt aktuelle Koordinaten
   yCrash=yCo-20;
   explosion ();                                                            //ruft "explosion auf
   treffer++;                                                               //erhöht Treffer um 1
   trefferAnzeige();                                                        //aktualisiert Trefferanzeige
  }



/*****************************************************************************
*              -1.Methode "explosion"-                                       *
* Methode, die die Explosionsanimation steuert(Lösung ohne Array oder Liste) *
*****************************************************************************/
/*  public void explosion () //*******************************************************
  {
   Graphics ge = getGraphics();
   ge.drawImage(img9,0,60,this);
   for (c=0 ;c<2  ;c++  )                                                   //Schleife wird 2 Mal durchlaufen
```

```
   {
    ge.drawImage(img10,(xCrash-30),(yCrash-30),this);speed.run(10);             //Zeichne Image und warte
    ge.drawImage(img11,(xCrash-30),(yCrash-30),this);speed.run(10);             //nächstes
    ge.drawImage(img12,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img13,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img14,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img15,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img16,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img17,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img18,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img17,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img16,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img15,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img14,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img13,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img12,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img11,(xCrash-30),(yCrash-30),this);speed.run(10);
    ge.drawImage(img10,(xCrash-30),(yCrash-30),this);speed.run(10);
   }
   ge.drawImage(img9,0,60,this);                                              //lösche ganzes Spielfeld
  }   */



/*****************************************************************************
*               -2.Methode "explosion"-                                      *
* Methode, die die Explosionsanimation steuert(Lösung mit Array )            *
*****************************************************************************/
  public void explosion () //*************************************************
  {
   Graphics ge = getGraphics();
   ge.drawImage(img9,0,60,this);                                              //Spielfeld neu zeichnen
   Image[] explosion = {img10,img11,img12,img13,img14,img15,img16,img17,img18}; //Array mit Bildern
   for(int c1=0;c1<2;c1++)                                                    //2X
    {int ex=0;                                                                //erstes Bild im Array
     for (c=0;ex<9;ex++)                                                      //erhöhe bis zum letzten Bild
      {ge.drawImage(explosion[ex],(xCrash-30),(yCrash-30),this);speed.run(10);} //zeige Bild 0-8,dazwischen warten
      ex--;                                                                   //letztes Bild
     for (c=0;ex>=0;ex--)                                                     //verringere bis zum ersten Bild
      {ge.drawImage(explosion[ex],(xCrash-30),(yCrash-30),this);speed.run(10);} //zeige Bild 8-0,dazwischen warten
    }
   ge.drawImage(img9,0,60,this);                                              //Spielfeld neu zeichnen
  }



/*****************************************************************************
*         Methode "ende"-                                                    *
* Methode, für Mausklick auf Ende-Button                                     *
*****************************************************************************/
```

```
  public void ende () //*****************************************************
  {
   if (xExit<392 && xExit>270 &&                                        //wenn Maus auf Button
       yExit<610 && yExit>560 &&
       pPoint==0 && nsPoint==0)                                         //und nicht Pause und nicht Neustart
         {
          System.exit(0);                                             //alles beenden
         }
  }



/*****************************************************************************
*                -Methode "neustart"-                                        *
* Methode, für Mausklick auf Start-Button                                    *
*****************************************************************************/
  public void neustart () //*****************************************************
  {
   if(xExit<128 && xExit>6 && yExit<610 && yExit>560 && pPoint==0)              //wenn Maus auf Button und nicht Pause
   {
    sound.intro();                                                          //Introsound
    nsPoint=1;                                                              //Maus wird blockiert
    t.stop();                                                               //beende aktuellen Thread
    Graphics ge = getGraphics();
    ge.setColor(Color.lightGray);                                           //zeichne Laden-Image
    ge.drawImage(img8,0,0,this);
    speed.run(1000);                                                        //warte
    setCursor(new Cursor(Cursor.WAIT_CURSOR));                              //setze Wait-Cursor
    int xe = 282;                                                           //zeichne Laden-Animation(Balken)
    for (c=0 ;c<142;c++)
         {
          ge.fillRect(xe,284,u,27);
          speed.run(20);
          u=u+1;
          xe++;
         }
    speed.run(1000);                                                        //warte
    System.gc();                                                            //Speicher räumen
    setCursor(new Cursor(Cursor.DEFAULT_CURSOR));                           //setze Standartcursor
    munition = mu;geschwindigkeit= ges;time=ti;leben = le;laenge = 1100;    //alle Variablen reset
    xCrash=0;yCrash=0;xExit=0;yExit=0;y=0;x=0;e=0;s=0;speedAnzeige=1;
    xCo=0;yCo=0;yPlus=0;yMinus=0;xPlus=0;xMinus=0;clear=0;clear1=100;
    clear2=100;click = 0;treffer=0;l=0;a=0;b=0;c=0;u=0;curserPointHand=1;
    curserPointKreuz=0;curserPointPfeil=0;xSchussAlt=0;schuss=0;strafpunkte=0;
    ufo=0;stern=0;komet=0;enterprise=0;lebenNeu=leben;nzGameOver=0;nzPause=0;
    nzStart=0;neuzeichnen=0;
    repaint();                                                              //neuzeichnen
   }
  }
```

```
/*****************************************************************************
*            -Methode "pause"-                                               *
* Methode, für Mausklick auf Pause-Button                                    *
*****************************************************************************/
  public void pause () //*************************************************
  {
   if (xExit<259 && xExit>138 &&                                    //Button Pause
       yExit<610 && yExit>560 &&
       pPoint==0 && nsPoint==0 && nzGameOver==0)                    //und nicht Pause,nicht Neustart,nicht GameOver
   {
    nzPause=1;                                                      //Neuzeichnen möglich
    pPoint=1;                                                       //Pausemarker auf 1
    Graphics ge = getGraphics();
    ge.drawImage(img23,0,0,this);
    t.suspend();                                                    //stope System
   }
   if (xExit<522 && xExit>401 && yExit<610 && yExit>560 && pPoint==1)   //Button Weiter
   {
    pPoint=0;                                                       //Pausemarker auf 0
    Graphics ge = getGraphics();
    ge.drawImage(img,0,0,this);
    ge.setFont(f1);
    ge.setColor(new Color(50,205,50));
    ge.drawString(" Treffer:   "+treffer,286,37);                   //Trefferanzeige wird aktualisiert
    ge.drawString("  Leben:   "+leben,417,37);                      //Lebensanzeige wird aktualisiert
    ge.setColor(new Color(220,20,60));
    ge.drawString(" Schuss:   "+munition,546,37);                   //Munitionsanzeige wird aktualisiert
    t.resume();                                                     //System weiter
    nzPause=0;                                                      //Neuzeichnen verhindern
   }
  }



/*****************************************************************************
*            -Methode "erstStart"-                                           *
* Methode, die nur beim ersten Start aufgerufen wird                         *
*****************************************************************************/
 public void erstStart () //*************************************************
  {
   if ((esPoint==0 && pPoint==0 && nsPoint==0) || nzStart==1)       //wenn nicht Erststart und nicht Pause und nicht Neustart
   {nzStart=1;                                                      //Neuzeichnen möglich
    sound.intro();                                                  //Introsound
    pPoint=1;                                                       //Pause blockieren
    Graphics ge = getGraphics();
    ge.setColor(Color.lightGray);
    ge.drawImage(img24,0,0,this);                                   //zeichne Info-Image
    speed.run(2000);                                                //warte
```

```
    ge.drawImage(img8,0,0,this);                                        //zeichne Start-Image
    speed.run(1000);                                                    //warte
    setCursor(new Cursor(Cursor.WAIT_CURSOR));                          //setze Wait-Cursor
    int xe1 = 282;
    for (c1=0 ;c1<142;c1++)                                             //zeichne Laden-Animation(Balken)
         {
          ge.fillRect(xe1,284,u,27);
          speed.run(20);
          u=u+1;
          xe1++;
         }
    munition = mu;geschwindigkeit= ges;time=ti;leben = le;laenge = 1100;        //alles reset
    xCrash=0;yCrash=0;xExit=0;yExit=0;y=0;x=0;e=0;s=0;speedAnzeige=1;
    xCo=0;yCo=0;yPlus=0;yMinus=0;xPlus=0;xMinus=0;clear=0;clear1=100;
    clear2=100;click = 0;treffer=0;l=0;a=0;b=0;c=0;u=0;curserPointHand=1;
    curserPointKreuz=0;curserPointPfeil=0;strafpunkte=0;lebenNeu=leben;
    speed.run(1000);                                                    //warte
    nzStart=0;                                                          //blockiere Neuzeichnen
    setCursor(new Cursor(Cursor.DEFAULT_CURSOR));                       //Standartcursor
    ge.drawImage(img25,0,0,this);                                       //zeichne Spielfeld-Image
    t.suspend();                                                        //stop

   }
   if((xExit<522&& xExit>401&& yExit<610&& yExit>560&& pPoint==1)|| nzStart==0) //Button Weiter und Pause blockiert
   {
    pPoint=0;                                                           //Pauseblockierung aufheben
    speed.run(1000);                                                    //warte
    nzStart=2;                                                          //Neuzeichnen blockieren
    t.resume();                                                         //weiter
   }
  }



/*****************************************************************************
*              -Methode "curserHand"-                                        *
* 1.Methode für den Cursor-Wechsel.Verantwortlich für Hand- und Standardcursor *
*****************************************************************************/
  public void cursorHand() //*****************************************************
  {
   if (((xCo<392 && xCo>270 && yCo<610 && yCo>560)||                    //wenn Maus auf einem Button
        (xCo<259 && xCo>138 && yCo<610 && yCo>560)||
        (xCo<128 && xCo>6 && yCo<610 && yCo>560)) &&
        ((curserPointHand==0 && curserPointKreuz==1)|| curserPointPfeil==1 ))
   {
    curserPointHand=1;                                                  //Marker setzen,die verhindern,dass
    curserPointKreuz=0;                                                 //ein gesetzter Cursor ständig neu
    curserPointPfeil=0;                                                 //gesetzt wird
    setCursor(new Cursor(Cursor.HAND_CURSOR));                          //setze Hand-Cursor
   }
```

```
    else  if(((xCo<786 && xCo>0 && yCo<52 && yCo>0)||                          //wenn Maus nicht auf Button oder Spielfeld
              (xCo<788 && xCo>402 && yCo<609 && yCo>560))&&
               curserPointPfeil==0 )
           {
            curserPointPfeil=1;                                               //Marker setzen
            setCursor(new Cursor(Cursor.DEFAULT_CURSOR));                     //Standartcursor setzen
           };
     cursorKreuz();                                                           //Methodenaufruf
   }



/*****************************************************************************
*                -Methode "curserKreuz"-                                     *
* 2.Methode für den Cursorwechsel.Verantwortlich für den Kreuz- und Standard-  *
* cursor.Diese könnte auch direkt in die erste Methode geschrieben werden, was *
* aber sehr unüberssichtlich ist                                              *
*****************************************************************************/
  public void cursorKreuz() //***************************************************
  {
   if(xCo<897 && xCo>0 && yCo<550 && yCo>54 &&                                //Wenn Maus im Spielfeld
     ((curserPointHand==1 && curserPointKreuz==0)|| curserPointPfeil==1))
   {
    curserPointKreuz=1;                                                       //Markerwechsel
    curserPointHand=0;
    curserPointPfeil=0;                                                       //setze Kreuzcursor
    setCursor(new Cursor(Cursor.CROSSHAIR_CURSOR));
   }                                                                          //wenn Maus nicht auf Button oder Spielfeld
   else  if(((xCo<786 && xCo>0 && yCo<52 && yCo>0)||
             (xCo<788 && xCo>402 && yCo<609 && yCo>560))&&
              curserPointPfeil==0)
           {
            curserPointPfeil=1;                                               //Markerwechsel
            setCursor(new Cursor(Cursor.DEFAULT_CURSOR));                     //Standartcursor setzen
           };
  }


/*****************************************************************************
*           -Methode "highscoreAnim"-                                       *
* Methode, die aufgerufen wird bei neuem Highscore.                           *
*****************************************************************************/
  public void highscoreAnim()//**************************************************
  {
   sound.highscore();                                                         //spiele Sound
   Graphics ge = getGraphics();
   ge.drawImage(img28,0,0,this);                                              //lade Bild
   for (c=0;c<1000;c++)                                                       //1000 x
   {
    ge.drawImage(img29,0,60,this);                                            //lade Bild
```

```
    speed.run(5);                                                   //warte
    ge.drawImage(img30,0,60,this);                                  //lade Bild
   }
   ge.drawImage(img3,0,0,this);                                     //lade Bild
   SpielPunkte sp =new SpielPunkte();                               //neue Instanz SpielPunkte
   sp.punkte(ge);                                                   //Methode Punkte
  }


}                                                                   //ENDE KLASSE ACTION




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                           --Klasse "Highscore"--                           *
* Hier wird der aktuelle Highscore aus einer Datei gelesen und mit dem Spiel- *
* stand verglichen. Wenn ein neuer Highscore erreicht wurde, wird die Datei   *
* überschrieben.                                                              *
*                                                                             *
*****************************************************************************/

import java.io.*;

public class Highscore extends SpielPunkte
{

 static FileReader f;
 static int c;                                                      //für f.read(char)
 static int w;                                                      //Summme aus eins,zehn,hundert
 static int eins = -1;                                              //Anfangsindex 3.Stelle
 static int zehn = -1;                                              //Anfangsindex 2.Stelle
 static int hundert = -1;                                           //Anfangsindex 1.Stelle



 public void lesen()
 {
  try {
         f = new FileReader("dat/highscore.dr");                    //Textdatei öffnen
         while ((c = f.read()) != -1) {                             //ein Zeichen(UNICODE)lesen
                                       w = c - '0';                 //macht aus c wieder int
                                       if(eins== -1 && zehn > -1){eins=w;}      //liest 3.Stelle
                                       if(zehn== -1 && hundert > -1){zehn=w;}   //liest 2.Stelle
```

```
                                    if(hundert== -1){hundert=w;}              //liest 1.Stelle
                                   }
        f.close();
      }
  catch (IOException e){}

  if(zehn>-1 && eins>-1){hundert=hundert*100;}                                 //wenn 2. und 3.Stelle dann hundert = 100
  if(zehn>-1 && eins==-1){hundert=hundert*10;}                                //wenn nur 2.Stelle dann hundert = 10
  if(eins>-1){zehn=zehn*10;}                                                  //wenn 1.Stelle dann zehn = 10
  if(eins==-1 && zehn>-1){zehn=zehn;}                                         //sonst zehn = 1
  if(eins==-1 && zehn==-1){zehn=0;}                                           //sonst 0
  if(eins==-1){eins=0;}                                                       //sonst 0
  w=hundert+zehn+eins;                                                        //Inhalt Texdatei als int
  eins = -1;                                                                  //reset
  zehn = -1;                                                                  //reset
  hundert = -1;                                                               //reset

 }




 public void schreiben()
 {
  String a = ""+total;                                                        //String aus int total (SpielPunkte)
  try
  {
   FileWriter output = new FileWriter("dat/highscore.dr");                    //neue Textdatei
   output.write(a);                                                           //schreiben
   output.close();                                                            //schliessen
  }
  catch (IOException e){}
 }

}




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                     --Klasse "NamenFrame"--                                *
* Hier weden Position und Grösse des Eingabefensters des Namen festgelegt.   *
*                                                                            *
*****************************************************************************/
```

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

class NamenFrame extends Frame
{



  public NamenFrame()
  {
    setTitle("SPACEMAN");                                                  //Titel
    setSize(950,65);                                                       //Groesse
    setLocation(0,-27);                                                    //Position
    addWindowListener(new WindowAdapter()
    {
      public void windowClosing(WindowEvent e)
      {
       //dispose();                                                        //schliessen (nicht möglich)
      }
    });

    add(new NamenPanel());                                                 //neues Panel
  }

}




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                    --Klasse "NamenPanel"--                                 *
* Hier wird der aktuelle Name des Highscore-Inhabers aus einer Datei gelesen   *
* und bei Bedarf überschrieben. Ausserdem wird hier bei Spielbeginn der Name   *
* der eingegeben wird, auf dem Fenster ausgegeben.                             *
*                                                                              *
*****************************************************************************/

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

class NamenPanel extends Panel  implements KeyListener
{
 static TextField tf4 = new TextField("",10);                              //leeres Textfeld
```

```
static char zeichen ;                                                   //Hilfs-Var.
static char namen[]= {' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '}; //Array der Länge 11 (gefüllt)
static int n=0;                                                         //Hilfs-Var.
static int m=0;                                                         //Hilfs-Var.
static int d;                                                           //Hilfs-Var.
static String na;                                                       //String für Namen (Eingabe)
static String na1;                                                      //String für Namen (Datei)
static int point = 0;                                                   //Hilfs-Var.
Font f = new Font("Comic Sans MS",Font.BOLD,20);
Font f1 = new Font("Comic Sans MS",Font.BOLD,25);



public NamenPanel()                                                     //Konstruktor
 {
   tf4.addKeyListener(this);                                            //Listener für Textfeld
   setLayout(new FlowLayout());                                         //Layout-Manager
   add(tf4);                                                            //Textfeld hinzufügen
 }



 public void paint(Graphics ge)                                         //Methode die zeichnet
 {
  if(n<11)                                                              //wenn Textfeld nicht voll
  {
   ge.fillRect(0,0,950,65);                                             //zeige Eingabeaufforderung
   ge.setColor(new Color(255,204,0));
   ge.setFont(f);
   ge.drawString("Bitte gib deinen Namen ein:",130,23);
   ge.drawString("Eingabe mit ENTER bestätigen !",540,23);
  }
  else {spielerNamen();}                                                //sonst zeige Eingabe
 }



 public void spielerNamen()                                             //Methode
 {
  Graphics ge = getGraphics();                                          //Hintergrundfarbe schwarz
  ge.fillRect(0,0,950,65);                                              //neue Farbe
  ge.setColor(new Color(255,204,0));
  ge.setFont(f);
  ge.drawString("Aktueller Spieler:",60,23);                            //zeige Eingabe
  ge.setColor(new Color(220,20,60));
  ge.setFont(f1);
  ge.drawString(" "+na,240,23);
 }
```

```
public void keyReleased(KeyEvent e){}                                   //nicht benötigt
public void keyPressed(KeyEvent e)
{point=1;
 if((e.getKeyCode() == KeyEvent.VK_BACK_SPACE) && n>0){n--;point=0;};    //wenn BachSpace-ein Zeichen zurück
 if(e.getKeyCode() == KeyEvent.VK_ENTER){n=11;};                        //wenn Enter-Feld voll
}



public void keyTyped(KeyEvent e)
{
 if(point==1)                                                           //wenn nicht BackSpace
 {
  zeichen = e.getKeyChar();                                             //lies Zeichen
  if(n<11){namen[n]=zeichen;n++;}                                       //wenn Feld nicht voll-Zeichen in Array
  else{ na=(""+namen[0]+namen[1]+namen[2]+namen[3]+namen[4]+             //wenn voll-Array in String
            namen[5]+namen[6]+namen[7]+namen[8]+namen[9]+namen[10]);
            spielerNamen();
            tf4.setVisible(false);                                      //Ausgabe auf Fenster
           }                                                            //Textfeld nicht mehr sichtbar
 }
}



public static void schreiben()                                          //Methode
 {
   try{

       FileWriter output = new FileWriter("dat/namen.dr");              //Datei öffnen
       output.write(na);                                                //schreiben
       output.close();                                                  //schliessen
      }
  catch (IOException e){}
 }



 public static void lesen()                                             //Methode
 {
   m=0;
   try {
        FileReader input = new FileReader("dat/namen.dr");              //Datei öffnen
        while ((d = input.read()) != -1) {                              //solange noch nicht leer
                                          namen[m]=(char)d;             //Zeichen in Array
                                          m++;                          //nächstes Feld im Array
                                         }
        input.close();                                                  //Datei schliessen
```

```
          na1=(""+namen[0]+namen[1]+namen[2]+namen[3]+namen[4]+                    //Array in String
               namen[5]+namen[6]+namen[7]+namen[8]+namen[9]+namen[10]);
          m=0;
         }
    catch (IOException e) {}
   }
}
```

```
// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                     --Klasse "Rahmen"--                                    *
* Hier weden Position und Grösse der einzelnen Teile des Aussenrahmens fest-  *
* gelegt und das Panel für den Aussenrahmen hinzugefuegt.                     *
*                                                                            *
*****************************************************************************/

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

 public class Rahmen extends Window
 {
    public void rahmen()                                                     //oben
    {
       final Rahmen fenster = new Rahmen();
       fenster.setLocation(0,38);                                            //Position
       fenster.setSize(950,25);                                              //Groesse
       fenster.setVisible(true);                                             //sichtbar
    }

    public void rahmen1()                                                    //unten
    {
       final Rahmen fenster = new Rahmen();
       fenster.setLocation(0,686);
       fenster.setSize(950,25);
       fenster.setVisible(true);
    }

    public void rahmen2()                                                    //links
    {
       final Rahmen fenster = new Rahmen();
       fenster.setLocation(0,38);
       fenster.setSize(25,653);
```

```
        fenster.setVisible(true);
    }

    public void rahmen3()                                                   //rechts
    {
        final Rahmen fenster = new Rahmen();
        fenster.setLocation(925,38);
        fenster.setSize(25,653);
        fenster.setVisible(true);
    }

    public Rahmen()
    {
        super(new Frame());                                                 //Konstrukter wird ueberschrieben
        add(new RahmenGrafik());                                            //Pabel wird hinzugefuegt
    }

}




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                    --Klasse "RahmenGrafik"--                               *
* Da das Hauptfenster von "Window" erbt ist es ohne Rahmen. Durch RahmenGrafik *
* erstellen wir ein eigenen Rahmen um das Hauptfenster. Da es sich nicht      *
* um einen echten Rahmen handelt, sondern um vier neue Fenster, die um das    *
* Hauptfenster positioniert werden, ist es möglich, ein eigenes Layout zu     *
* erstellen.                                                                  *
*                                                                             *
*****************************************************************************/

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

class RahmenGrafik extends Panel
 {                                                                          //Grafik für den Aussenrahmen
  Image rahmen;

  public RahmenGrafik()
  {
   MediaTracker mt = new MediaTracker(this);                                //Rahmengrafik wird geladen
   rahmen = Toolkit.getDefaultToolkit().
   getImage("pics/rahmen1.jpg");
```

```
   mt.addImage(rahmen,0);
   try{mt.waitForID(0);}
   catch(InterruptedException ie) { repaint();}
  }

  public void paint(Graphics g)
  {
   g.drawImage(rahmen,0,0,this);                                    //wird gezeichnet
  }

 }




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                  --Klasse "Sound"--                                        *
* Die Klasse sorgt mit seinen Methoden für die nötigen Soundeffekte und Hinter-*
* grundmusik. Der Soundeffekt "Feuer" läuft als Thread in der Methode "run",   *
* die durch "start" aufgerufen wird. Dies ist notwendig, damit der Spielablauf *
* während des Abspielens der Sounddatei nicht unterbrochen wird. In der        *
* Methode "background" wird die Hintergrundmusik als Endlosschleife durch Auf- *
* ruf von "loop" abgespielt.                                                   *
*                                                                              *
*****************************************************************************/

import java.net.*;
import java.applet.*;

class Sound implements Runnable
{
  Thread t2;

 public void start()
 {
  t2 = new Thread(this);                                            //neuer Thread
  t2.start();                                                       //ruft run als Thread auf
  }

 public void run()                                                  //läuft als Thread
 {
  try {                                                             //spielt Sound "Feuer" 1x
       URL url = new URL("file:"+
                         System.getProperty("user.dir")+
```

```
                        System.getProperty("file.separator")+
                        "sound/feuer.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void getroffen (  )
{                                                                    //spielt Sound "getroffen" 1x
 try {
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/getroffen.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void intro()
{
 try {                                                               //spielt Sound "intro" 1x
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/intro.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void gameOver()
{                                                                    //spielt Sound "gameover" 1x
 try {
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/gameover.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void uhr()
{
 try {                                                               //spielt Sound "uhr" 1x
```

```
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/uhr.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void explosion()
{
 try {                                                          //spielt Sound "explosion" 1x
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/explosion.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}

public void background()
{                                                               //spielt Sound "background"
 try {                                                          //als Schleife
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/background.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.loop();
      }
 catch (MalformedURLException e) {}
}


 public void highscore()
{
 try {                                                          //spielt Sound "highscore" 1x
      URL url = new URL("file:"+
                        System.getProperty("user.dir")+
                        System.getProperty("file.separator")+
                        "sound/highscore.wav");
        AudioClip clip = Applet.newAudioClip(url);
        clip.play();
      }
 catch (MalformedURLException e) {}
}
```

```
}
```

```
// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                          --Klasse "Speed"--                                *
* Eigentlich überfluessig, da die Methode "sleep" der Klasse "Thread" direkt  *
* aufgerufen werden kann. Da "sleep" aber nur sehr unzuverlaessig arbeitet,   *
* wurde sie in eine eigene Klasse integriert, um später eine Änderung vornehmen*
* zu können.                                                                  *
* Funktion: den Int-Wert "zeitindex" der übergeben wird, wartet das System in *
* Millisekunden. Auf diese Weise kann die Geschwindigkeit der Ablaufe beein-  *
* flusst werden.                                                              *
*                                                                             *
*****************************************************************************/

class Speed extends Thread
{
 public void run(int zeitindex)
 {
  try {
       sleep(zeitindex);
      }
        catch(InterruptedException e)
        {
        }

 }
}
```

```
// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*           --Klasse "SpielPunkte"--                                          *
* Die Klasse enthaelt nur die Methode Punkte, in der der Spielstand errechnet *
* wird und eine Ausgabe auf dem Spielfeld erfolgt. Sie erbt unmittelbar von   *
```

```
* "Action".                                                                    *
*                                                                             *
*******************************************************************************/

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

class SpielPunkte extends Action
{
 static int ufoPunkte, sternPunkte, kometPunkte, strafPunkteNeu, total, zeitbonus;     //Punkte nach jedem Spiel

 public void punkte(Graphics ge)
  {
   ufoPunkte =ufo*3;                                                          //alle Treffer auf Ufo*3
   sternPunkte =stern*2;                                                      //alle Treffer auf Stern*2
   kometPunkte =komet*2;                                                      //alle Treffer auf Komet*2
   zeitbonus=lebenNeu*3;                                                      //Zeitbonus ist Anzahl der Leben*3
   strafPunkteNeu =(-strafpunkte);                                            //Strafpunkte
   total=ufoPunkte+sternPunkte+kometPunkte+enterprise+zeitbonus+strafPunkteNeu; //Gesamtpunktzahl wird addiert
   ge.setColor(new Color(255,204,0));                                         //Ausgabe
   ge.setFont(f2);                                                            //dto
   ge.drawString(""+(ufo),480,240);                                           //dto
   ge.drawString(""+(stern),480,282);                                         //dto
   ge.drawString(""+(komet),480,324);                                         //dto
   ge.drawString(""+enterprise,480,366);                                      //dto
   ge.drawString(""+lebenNeu,480,409);                                        //dto
   ge.drawString(""+strafPunkteNeu,480,455);                                  //dto
   ge.setColor(new Color(178,34,34));                                         //dto
   ge.drawString(""+total,480,507);                                           //dto
  }
}




// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*******************************************************************************
*                            --Klasse "StarteGame"--                           *
* Das Fenster wird erzeugt. Da dieses von "Window" erbt, hat es keinen Rahmen. *
* Der Aussenrahmen wird dann durch die Methoden Rahmen() bis Rahmen3() der     *
* Klasse Rahmen erzeugt. Der Konstruktor wird ueberschrieben und das Panel     *
* hinzugefuegt.                                                               *
*                                                                             *
*******************************************************************************/
```

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

public class StarteGame extends Window
{
   public static void main(String[] args)
   {
      Rahmen ra = new Rahmen();
      ra.rahmen();                                                    //Aussenrahmen
      ra.rahmen1();                                                   //dto
      ra.rahmen2();                                                   //dto
      ra.rahmen3();                                                   //dto

      Sound d = new Sound();
      d.background();

      final StarteGame fenster = new StarteGame();                    //Hauptfenster
      fenster.setLocation(25,63);                                     //Position
      fenster.setSize(900,623);                                       //Grösse
      fenster.setVisible(true);                                       //sichtbar

   }

   public StarteGame()
   {
    super(new Frame());                                               //Konstruktor wird überschrieben
    add(new Action());                                                //Panel wird zum Rahmen hinzugefuegt
   }
}
```

```java
// SPACEMAN
// v.0.1
// @author: D.Reinhold / J.Strehmann
// Datum: März 2004

/*****************************************************************************
*                    --Klasse "Welt"--                                       *
* Die Klasse enthält alle benötigten Variablen und vererbt diese an die Klasse *
* "Action". Im Konstruktor werden alle Bilder in den Speicher geladen um eine  *
* fluessige Bewegung zu garantieren.                                          *
*                                                                             *
*****************************************************************************/

import java.awt.*;
import java.awt.event.*;
public class Welt extends Panel
```

```
{
 int munition = 20;      int mu=munition;                                    //doppelte Zuweisung für neustart()
 int geschwindigkeit=7; int ges=geschwindigkeit;                             //dto
 int time=200;           int ti=time;                                        //dto
 int leben = 7;          int le=leben;

 Image img,img1,img2,img3,img4,img5,img6,img7,img8,img9,img10,img11,          //Grafiken
       img12,img13,img14,img15,img16,img17,img18,img19,img20,img21,
       img22,img23,img24,img25,img26,img27,img28,img29,img30,feind;

 int xSchussAlt;                                                             //Var für Schusszahl
 int schuss;                                                                 //dto
 int curserPointHand=1;                                                      //Pointer für Cursorwechsel
 int curserPointKreuz=0;                                                     //dto
 int curserPointPfeil=0;                                                     //dto
 int xKreuzNeu,yKreuzNeu,xKreuzAlt,yKreuzAlt;                                //Var. für kreuzNeu()
 int lPoint;                                                                 //Pointer für Laden nach Erststart
 int esPoint;                                                                //Pointer für Erststart
 int sPoint;                                                                 //Pointer für Aufprall
 int pPoint;                                                                 //Mausaktion bei Pause
 int nsPoint;                                                                //Mausaktion bei Neustart
 int laenge = 1100;                                                          //Laenge der Flugbahn der Objekte
 int xCrash;                                                                 //Hilfsvariable, damit bei Mausbewegung
                                                                             //Crash-Animation am Platz bleibt
 int yCrash;                                                                 //dto
 int xExit;                                                                  //X-Koord. für Ende
 int yExit;                                                                  //Y-Koord. für Ende
 int y;                                                                      //Random Y-Koord. für Flugobjekte
 int x;                                                                      //Random X-Koord. für Flugobjekte
 int e;                                                                      //Zähler Zeit auf 0
 int s;                                                                      //Zähler Geschwindigkeit auf 0
 int speedAnzeige=1;                                                         //Anzeige Geschwindigkeit
 int xCo;                                                                    //Koordinate
 int yCo;                                                                    //dto
 int yPlus;                                                                  //Zuweisung Koordinate
 int yMinus;                                                                 //dto
 int xPlus;                                                                  //dto
 int xMinus;                                                                 //dto
 int clear;                                                                  //alte Koordinate
 int clear1=100;                                                             //dto
 int clear2=100;                                                             //dto
 int click = 0;                                                              //Pointer für Mausklick
 int treffer;                                                                //Trefferanzahl
 int tre;                                                                    //Pointer für Treffer
 int c;                                                                      //for-Schleifen Index
 int c1;                                                                     //for-Schleifen Index
 int u;                                                                      //for-Schleifen Index
 int l;                                                                      //Zähler Leben
 int neuzeichnen=0;                                                          //Neuzeichnen bei Überdeckung
```

```
int nzStart=0;                                                  //Neuzeichnen bei Start
int nzPause=0;                                                  //Neuzeichnen bei Pause
int nzGameOver=0;                                               //Neuzeichnen bei Game Over
double a;                                                       //Random-Zahl
double b;                                                       //dto.
Thread t;                                                       //Thread der Paint-Methode
Font f1 = new Font("Sanserif",Font.BOLD,15);                    //Schriftfont
Font f2 = new Font("Comic Sans MS",Font.BOLD,30);               //dto
Font f3 = new Font("Comic Sans MS",Font.BOLD,20);               //dto

public Welt()
 {
   MediaTracker mt = new MediaTracker(this);

   img = Toolkit.getDefaultToolkit().
   getImage("pics/welt.jpg");
   mt.addImage(img,0);

   img1 = Toolkit.getDefaultToolkit().
   getImage("pics/ufo.jpg");
   mt.addImage(img1,0);

   img2 = Toolkit.getDefaultToolkit().
   getImage("pics/komet.jpg");
   mt.addImage(img2,0);

   img3 = Toolkit.getDefaultToolkit().
   getImage("pics/gameover.jpg");
   mt.addImage(img3,0);

   img4 = Toolkit.getDefaultToolkit().
   getImage("pics/ufo1.jpg");
   mt.addImage(img4,0);

   img5 = Toolkit.getDefaultToolkit().
   getImage("pics/clear.jpg");
   mt.addImage(img5,0);

   img6 = Toolkit.getDefaultToolkit().
   getImage("pics/kreuz.gif");
   mt.addImage(img6,0);

   img7 = Toolkit.getDefaultToolkit().
   getImage("pics/kreuz1.gif");
   mt.addImage(img7,0);

   img8 = Toolkit.getDefaultToolkit().
   getImage("pics/start.jpg");
   mt.addImage(img8,0);
```

```
img9 = Toolkit.getDefaultToolkit().
getImage("pics/sterne.jpg");
mt.addImage(img9,0);

img10 = Toolkit.getDefaultToolkit().
getImage("pics/crash0.jpg");
mt.addImage(img10,0);

img11 = Toolkit.getDefaultToolkit().
getImage("pics/crash1.jpg");
mt.addImage(img11,0);

img12 = Toolkit.getDefaultToolkit().
getImage("pics/crash2.jpg");
mt.addImage(img12,0);

img13 = Toolkit.getDefaultToolkit().
getImage("pics/crash3.jpg");
mt.addImage(img13,0);

img14 = Toolkit.getDefaultToolkit().
getImage("pics/crash4.jpg");
mt.addImage(img14,0);

img15 = Toolkit.getDefaultToolkit().
getImage("pics/crash5.jpg");
mt.addImage(img15,0);

img16 = Toolkit.getDefaultToolkit().
getImage("pics/crash6.jpg");
mt.addImage(img16,0);

img17 = Toolkit.getDefaultToolkit().
getImage("pics/crash7.jpg");
mt.addImage(img17,0);

img18 = Toolkit.getDefaultToolkit().
getImage("pics/crash8.jpg");
mt.addImage(img18,0);

img19 = Toolkit.getDefaultToolkit().
getImage("pics/munition.jpg");
mt.addImage(img19,0);

img20 = Toolkit.getDefaultToolkit().
getImage("pics/stern.jpg");
mt.addImage(img20,0);

img21 = Toolkit.getDefaultToolkit().
getImage("pics/enterprise.jpg");
```

```
    mt.addImage(img21,0);

    img22 = Toolkit.getDefaultToolkit().
    getImage("pics/kollision.jpg");
    mt.addImage(img22,0);

    img23 = Toolkit.getDefaultToolkit().
    getImage("pics/pause.jpg");
    mt.addImage(img23,0);

    img24 = Toolkit.getDefaultToolkit().
    getImage("pics/info.jpg");
    mt.addImage(img24,0);

    img25 = Toolkit.getDefaultToolkit().
    getImage("pics/anweisung.jpg");
    mt.addImage(img25,0);

    img26 = Toolkit.getDefaultToolkit().
    getImage("pics/feuer.gif");
    mt.addImage(img26,0);

    img27 = Toolkit.getDefaultToolkit().
    getImage("pics/feuer1.gif");
    mt.addImage(img27,0);

    img28 = Toolkit.getDefaultToolkit().
    getImage("pics/highscore1.jpg");
    mt.addImage(img28,0);

    img29 = Toolkit.getDefaultToolkit().
    getImage("pics/highscore2.gif");
    mt.addImage(img29,0);

    img30 = Toolkit.getDefaultToolkit().
    getImage("pics/highscore3.gif");
    mt.addImage(img30,0);

     try{mt.waitForID(0);}
     catch(InterruptedException ie) {}
  }
}
```