



Mehr Spielspaß mit Java auf Homepages

Einführung in Java



Wollen Sie die **Besucher Ihrer Homepage** überraschen? Dann nutzen Sie **Java**, die Plattform unabhängige **Programmiersprache**. Nach einer **kurzen Einführung** können Sie **Spiele** selbst entwickeln.

ROLAND WILLMS

Java gibt es offiziell seit dem 23.05.1995. Java ist eine mächtige Programmiersprache, aber die Grundregeln sind spielend leicht zu

QUICK INDEX

- **Installation des Java 2 SDK**
Das kostenlose Java 2 SDK liefert alle Tools zur Entwicklung von Java-Programmen.
- **Applets auf Webseiten**
Das Spiel mit der herumspringenden Maus ist auf einer Webseite eingebettet.
- **Der Umgang mit Objekten**
Als Modell für die Kommunikation zwischen Objekten ist ein Kampf zwischen Hund und Katze gut geeignet.
- **Mit primitiven Werten rechnen**
Auf Wahrheitswerte, Zahlen und Zeichen kann kein Programm verzichten.
- **Baupläne für Objekte**
Eine Klasse legt den Zustand und das Verhalten von Objekten eines bestimmten Typs fest.
- **Zur Applikation übergehen**
Programme können als Applet und als Applikation ablaufen.
- **Die Tools des Java 2 SDK**
Den Compiler, den Interpreter und den Applet Viewer sollte jeder Entwickler kennen.

verstehen. Der Name bedeutet umgangssprachlich "Kaffee". Vielleicht landen Sie in den USA mal in einem "Java Shop".

■ Yellow Dog Linux Champion Server 1.2.1

Das Java 2 SDK (Software Development Kit) ist im Internet kostenlos erhältlich. <http://java.sun.com/j2se/>

Nach einem Doppelklick auf die EXE-Datei startet die Installation. Als Zielordner ist C:\jdk1.3 voreingestellt. Auch die Dokumentation zum Java API (Application Programming Interface) ist sehr zu empfehlen. Der Ordner C:\jdk1.3\bin muss im Suchpfad PATH stehen, um mit den Tools arbeiten zu können. Unter Windows 95 / 98 schreiben Sie die Zeile

```
set PATH=C:\jdk1.3\bin;%PATH%
```

ans Ende der Datei autoexec.bat. Unter Windows NT 4 / 2000 wählen Sie die Einträge "Einstellungen / Systemsteuerung" im Startmenü aus, öffnen das Fenster "Systemeigenschaften" über das Symbol "System", wechseln zur Registerkarte "Erweitert" und klicken auf die Schaltfläche "Umgebungsvariablen". Nun wird die Variable PATH über "Neu" eingerichtet oder über "Bearbeiten" angepasst.

Fremde Programme haben manchmal die Variable CLASSPATH eingerichtet, sodass ein NoClassDefFoundError auftritt. Erweitern Sie ihren Wert einfach um einen Punkt, damit Klassen im gegenwärtigen Arbeitsordner gesucht werden können.

■ Applets auf Webseiten

Als praktisches Beispiel wird das Spiel JumpingMouse entwickelt. Die heutigen Browser "Microsoft Internet Explorer 5", "Netscape 6" und "Opera 5" zeigen die Webseite JumpingMouse.html problemlos an.

Der Spieler hat die Aufgabe, sich mit der Katze langsam an die Maus heranzuschleichen und sie durch blitzschnelle Klicks zu fangen. Die Maus beobachtet die Katze ständig und springt auf der Wiese lachend hin und her.

Bei den Java-Programmen unterscheidet man zwei Typen. Applets liegen auf einer Webseite, so dass sie in einem Browser ablaufen. Niemand kann sich sicher sein, ob ein Programm friedlich ist, das von einem fremden Rechner im Internet stammt. Daher gibt es Einschränkungen, um die Sicherheit des Betriebssystems zu wahren. Applikationen hingegen sind normale Programme, die in einer Eingabeaufforderung gestartet werden. Sie dürfen auf die Systemressourcen uneingeschränkt zugreifen. Jeder Benutzer muss für sich selbst entscheiden, ob er dem Entwickler vertraut, falls er ihn nicht kennt.

Sehen Sie sich nun die Webseite JumpingMouse.html in einem Texteditor an. Einige Grundkenntnisse in HTML sind nötig, um alle Details zu verstehen. Wichtig ist das APPLET-Tag zur Einbettung des Spiels.



JUMPING MOUSE als Applet, es gibt sie auch als Applikation.

```
<APPLET
code="JumpingMouse.class"width="300
" height="300">
</APPLET>
```

Aus den Attributen code, width und height ergibt sich, dass der Bytecode des Applets in der Datei JumpingMouse.class liegt und die gelbliche Spielwiese eine Größe von 300 x 300 Pixel hat.

■ Der Umgang mit Objekten

Eine Klasse ist ein Bauplan, in dem die Eigenschaften von Objekten eines bestimmten Typs festgelegt sind. Zur Visualisierung solcher Pläne gibt es UML-Diagramme (Unified Modeling Language). Das Diagramm für eine Klasse besteht aus drei Zellen mit Namen (obere Zelle), Feldern für den Zustand (mittlere Zelle), Konstruktoren zur Erschaffung von Exemplaren und den Methoden für das Verhalten (untere Zelle). In der Oberklasse Beast finden Sie die gemeinsamen Eigenschaften von Cat und Dog. An die Pfeile können Sie in Gedanken "erbt Eigenschaften von" schreiben.

Die Antworten auf die drei Fragen "Was ist eine Katze?", "Was hat eine Katze?" und "Was macht eine Katze?" sind nicht schwer. Eine Katze ist ein Raubtier, das eine Anzahl teeth von Zähnen, eine Länge length und eine Fellfarbe

skin hat. Mit der Methode fight kann sie einen Hund bekämpfen.

Konstruktoren sind Hilfsmittel zur Erschaffung von Objekten. Ein Beispiel: Der Konstruktor für eine Katze verlangt in den runden Klammern einige Argumente. Hinter den Typen int, double und Color verbirgt sich eine Ganzzahl, eine Fließkommazahl und eine Farbe. Eine Katze mit 12 Zähnen, der Länge 41.5 cm und einem grauen Fell wird mit dem Ausdruck

```
new Cat(12,41.5,
Color.gray)
```

geboren. Um sie später ansprechen zu können, erhält sie den Namen pussy. In der Zeile

```
Cat pussy = new Cat(12,
41.5, Color.gray);
```

wird die Variable pussy des Typs Cat eingerichtet, die über den Operator = einen vernünftigen Wert erhält. Der Hund doggy hat ein paar Zähne mehr und ist etwas länger.

```
Dog doggy = new Dog(24, 62.7, new
Color(128, 64, 0));
```

Die Farbanteile 128, 64 und 0 sorgen für ein braunes Fell.

In Dog steht die Methode bite, um eine Katze zu beißen. Der Täter kommt vor die Methode und das Opfer in die runden Klammern. Die Anweisung

```
doggy.bite(pussy);
```

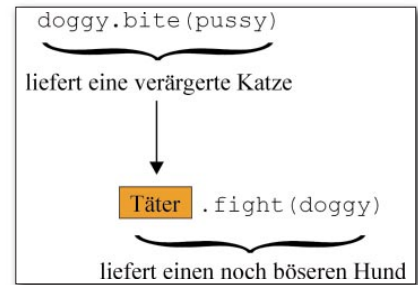
sorgt dafür, dass der Hund die Katze beißt. Die verärgerte Katze reagiert sofort mit der Zeile

```
pussy.fight(doggy);
```

auf die Aktion des Hundes. Manchmal liefern Methoden ein Ergebnis. Zum Beispiel steht der Typ Cat hinter dem Doppelpunkt bei der Methode bite. Als Ergebnis wird die verärgerte Katze zurückgegeben, die Sie in der letzten Anweisung anstelle von pussy eintragen können.

```
doggy.bite(pussy).fight(doggy);
```

Jetzt sind zwei Methoden verkettet. Mit dem Feld teeth erfahren Sie, ob die Katze während des Kampfes einen Zahn verloren hat.



ERGEBNISSE direkt wiederverwerten.

```
int vorher = pussy.teeth;
doggy.bite(pussy).fight(doggy);
int nachher = pussy.teeth;
```

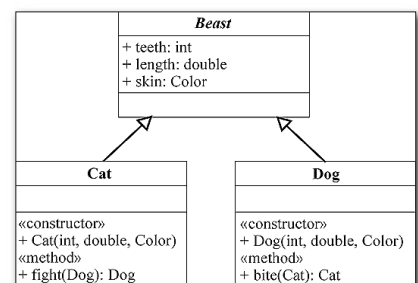
Die Anzahl der verlorenen Zähne ergibt sich aus der Differenz vorher – nachher.

■ Mit primitiven Werten rechnen

Java unterscheidet primitive und referenzierende Typen. Die Wahrheitswerte des Typs boolean, die Ganzzahlen der Typen byte, short, int und long, die Fließkommazahlen der Typen float und double und die Zeichen des Typs char sind primitiv. Die einzigen Wahrheitswerte sind false und true. Im Bereich von -2.147.483.648 bis 2.147.483.647 liegen die Ganzzahlen des Typs int, was in der Praxis häufig ausreicht. Der bekannteste Typ für Fließkommazahlen zwischen 4.9E-324 und 1.8E+308 ist double. Hinter dem E steht die Zehnerpotenz.

Bei einem Zeichen handelt es sich eigentlich um eine Ganzzahl im Bereich von 0 bis 65.535. Weil die Ermittlung der Platznummer im Unicode zu umständlich ist, wird es in zwei einfache Hochkommata eingeschlossen, zum Beispiel 'A'.

TIPP Referenzierende Werte sind Objekte, deren Zustand oft so vielfältig ist, dass sie nur mit Konstruktoren erschaffen werden. Der Typ String für Zeichenketten bildet die einzige Ausnahme. Hier schließen Sie einfach ein paar Buchstaben in zwei doppelte Hochkommata ein, zum Beispiel "Hallo".



UML-DIAGRAMME von Beast, Cat und Dog



HUND UND KATZE kommunizieren.

Für die Spielwiese mit der Maus und der Katze benötigen Sie eine Grafikscreen. In der Klasse Graphics gibt es die Methoden setColor(Color) zum Einstellen einer Malfarbe, fillRect(int, int, int, int) zum Füllen eines Rechtecks und drawImage(Image, int, int, ImageObserver) zur Ausgabe eines Bildes. Werfen Sie zunächst einen Blick über das folgende Listing.

```
screen.setColor(new Color(255,
255, 202));
screen.fillRect(0, 0, 300, 300);
if (over) {
    screen.setColor(Color.black);
    screen.fillRect(jerryX - 12,
jerryY - 34, 10, 70);
    screen.fillRect(jerryX - 32,
jerryY - 14, 50, 10);
} else {
    screen.drawImage(jerry, jerryX
- 32, jerryY - 34, this);
}
if (inside) {
    screen.drawImage(tom, tomX -
45, tomY - 55, this);
}
```

Der Konstruktor Color(int, int, int) erhält die Anteile 255, 255 und 202 der gelblichen Wiesenfarbe. Zum Füllen eines Rechtecks geben Sie die Koordinaten (0, 0) der linken oberen Ecke, die Breite 300 und die Höhe 300 an. Von der Variablen over des Typs boolean hängt der weitere Verlauf der Anweisungen ab.

Beim Wert true ist das Spiel vorbei. Als Malfarbe wird nun Schwarz genommen, sodass die beiden Methoden zum Füllen eines Rechtecks für ein schwarzes Kreuz beim Ort der Maus sorgen. Beim Wert false erscheint das Bild jerry der Maus anstelle des Kreuzes. jerryX und jerryY sind die Koordinaten des Mittelpunktes der Maus, so dass die Hälfte der Breite 64 und der Höhe 68 des Bildes abzuziehen ist. Zur Platzierung von Bildern geben Sie die Koordinaten der linken oberen Ecke an. Als ImageObserver dient das Applet JumpingMouse, so dass der Täter this als

Hinweis auf das gegenwärtige Objekt übergeben wird.

Die Variable inside hat ebenfalls den Typ boolean. Wenn die Katze mit den Koordinaten tomX und tomY ihres Mittelpunktes auf der Wiese ist, erscheint das Bild tom. Neben if Anweisungen gibt es in Java noch weitere Kontrollstrukturen, zum Beispiel for, while, do-while und switch. Die Transferanweisungen break; und continue; erlauben ein vorzeitiges Abbrechen oder Wiederholen einer Schleife.

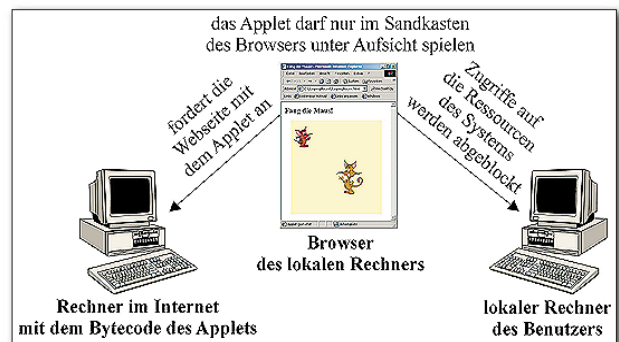
Die Klasse Color hat statische Felder mit vordefinierten Farben, wie black. In UML-Diagrammen sind statische Eigenschaften unterstrichen. Wenn das Feld teeth eines Raubtiers statisch wäre und die Katze pussy einen Zahn verlieren würde, hätten auch alle anderen Katzen automatisch einen Zahn weniger. Weil die Hunde das Feld teeth erben, würden auch sie sich über einen fehlenden Zahn wundern. Statische Eigenschaften sind für alle Objekte eines Typs gleich, so dass Sie den Klassennamen als Täter nehmen, zum Beispiel Color.black.

Es gibt zahlreiche Operatoren zur Ver-

arbeitung primitiver Werte. Häufig tauchen bei Wahrheitswerten ! (Negation), & (logisches Und), | (logisches Oder) und bei Zahlen + (Addition), - (Subtraktion), * (Multiplikation), / (Division), % (Restbestimmung), ++ (Erhöhung um 1), - (Erniedrigung um 1) auf. Zum Vergleichen oder Anordnen dienen == (gleich), != (ungleich), < (kleiner), <= (kleiner oder gleich), > (größer), >= (größer oder gleich).

Referenzierende Werte kommunizieren untereinander mit Methoden. Als einzige Ausnahme dürfen Sie mehrere Zeichenketten mit + aneinander hängen. Es ist erlaubt, eine Zeichenkette und einen primitiven Wert zusammenzufügen.

In vielen Fällen ist es praktisch, eine Entwicklungsumgebung zur Hand zu haben, die schnell Anweisungen in Java auf Knopfdruck ausführen kann. Den JLauncher installieren Sie durch Ent-



APPLETS SPIELEN im Sandkasten unter Aufsicht.

Umgang mit Objekten:

Feldabfrage: **Täter** . **Feld**

Objekterschaffung: **new** ↑ **Konstruktor** (**Liste mit Werten**)
Leerzeichen

Methodenaufruf: **Täter** . **Methode** (**Liste mit Opfern**)

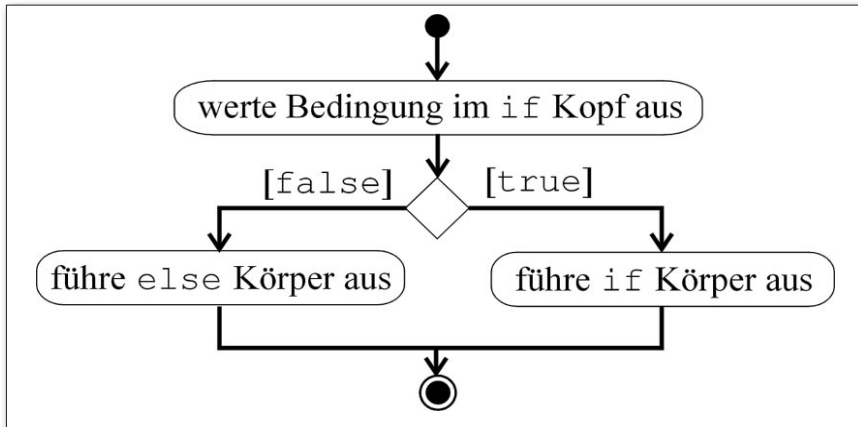
Einrichtung von Variablen:

Deklaration: **Typ** ↑ **Bezeichner** ;
Leerzeichen

Zuweisung: **Bezeichner** ↑ = ↑ **Wert** ;
Leerzeichen

Initialisierung: **Typ** ↑ **Bezeichner** ↑ = ↑ **Wert** ;
Leerzeichen

MIT OBJEKTEN und Variablen umgehen, Objekte erschaffen, Variablen einrichten.



FLUSSDIGRAMM für eine Bedingung, in Java wie anderswo.

packen des Archivs, das auf der Homepage <http://www.jlauncher.com/> zum Download angeboten wird. Nach dem ersten Start rufen Sie das Menü "Einstellungen / Konfiguration" auf und tragen die Befehle javac, java, appletviewer und javadoc in die Textfelder ein.

Die Klasse System hat einen statischen Druckstrom out. Die Methode print dient zum Ausdrucken von Ergebnissen beliebigen Typs auf der Standardausgabe. Öffnen Sie im JLauncher über das Menü "Arbeitsblatt / Neu" ein Arbeitsblatt für eine Applikation. Um das Ergebnis von $1 + 2 + 3$ zu erhalten, führen Sie die Anweisung

```
System.out.print(1 + 2 + "3");
```

über das Menü "Bearbeiten / Ausführen" aus. Ausdrücke werden von links nach rechts ausgewertet. Bei $1 + 2$ ergibt sich 3, so dass bei $3 + "3"$ die Zeichenkette 33 entsteht, die im Textbereich "Meldungen" erscheint.

Baupläne für Objekte

Der Quellcode der Klasse JumpingMouse startet mit ihrer Deklaration.

```
public class JumpingMouse extends
Applet implements MouseListener,
MouseMotionListener {
    <Felder, Konstruktoren und
    Methoden>
}
```

Hinter extends stehen die Oberklasse und hinter implements zwei Schnittstellen, um auf Ereignisse zu reagieren.

In Java darf eine Klasse nicht zwei Oberklassen haben. Weil ein Hund bereits ein Raubtier ist, sind dressierte Hunde nur möglich, wenn Sie die Schnittstelle Training implementieren. Schnittstellen sind Sammlungen mit statischen Feldkonstanten und abstrakten Methoden, die in Unterklassen über-

schrieben und mit Anweisungen gefüllt werden.

Vor öffentlichen Eigenschaften steht in UML-Diagrammen ein Plus-Zeichen, vor privaten ein Minus-Zeichen. Private Eigenschaften dürfen Sie nur innerhalb der Klasse nutzen. Die Deklaration eines Feldes besteht oft aus private, einem Typ und einem Bezeichner, zum Beispiel

```
private boolean insi-
de;
```

Alle Felder haben automatisch Anfangswerte. Inside hat den Wert false und jerryX den Wert 0. Mit einer Zuweisung hinter der Deklaration lässt sich das schnell ändern.

```
private int jerryX =
150;
```

Am Anfang wartet die Maus in der Mitte der Spielwiese auf die Katze. Es ist nicht üblich, einen Konstruktor in ein Applet einzufügen. Ein Standardkonstruktor wird automatisch vorgesehen.

Nachdem der Browser ein Exemplar des Spiels erschaffen hat, ruft er die Methode init auf. Hier werden zum Beispiel die Bilder für die Maus und die Katze geladen.

```
public void init() {
    jerry =
getImage("Jerry.gif");
    tom =
getImage("Tom.gif");
    <weitere Anweisun-
    gen>
}
```

Die Deklaration einer Methode besteht aus private

oder public, dem Typ des Ergebnisses, einem Bezeichner und eventuell einer Liste mit Argumenten in den runden Klammern. Methoden, die kein Ergebnis liefern, kennzeichnen Sie durch void. Für die Spielwiese erschaffen Sie ein Bild.

```
buffer = createImage(300, 300);
screen = buffer.getGraphics();
```

Auf der Grafik screen finden die besprochenen Malereien statt. Die vererbte Methode update wird überschrieben, um ein Flackern beim Herumspringen der Maus zu vermeiden.

```
public void update(Graphics g) {
    paint(g);
}
```

Bei jedem Argument geben Sie den Typ und einen Namen an. Die übergebenen Werte stehen im Rumpf der Methode unter diesen Namen zur Verfügung. In die Methode paint kommen die besprochenen Malanweisungen.

```
public void paint(Graphics g) {
    <Malanweisungen>
}
```



ZAHLENSPIELE im JLauncher.

DIE HTML-SEITE

```
<HTML>
<HEAD>
    <TITLE>Fang die Maus!</TITLE>
</HEAD>
<BODY>
    <H2>Fang die Maus!</H2>
    <CENTER>
        <APPLET code="JumpingMouse.class"
        width="300" height="300"></APPLET>
    </CENTER>
</BODY>
</HTML>
```



DIE LISTINGS...

...gibt es unter www.pc-magazin.de/download/special_21/

```
g.drawImage(buffer, 0, 0,
this);
}
```

Auf die Grafik g des Applets malen Sie die Spielwiese buffer. Die beiden implementierten Zuhörer sorgen für die Reaktion auf Ereignisse, die bei den Bewegungen und den Klicks der Katze entstehen. Zunächst registrieren Sie sie in der Methode init.

```
addMouseListener(this);
addMouseMotionListener(this);
```

Anschließend müssen die sieben Methoden sinnvoll überschrieben werden. Methoden ohne Bedeutung bleiben leer, zum Beispiel

```
public void mouseReleased(
MouseEvent e) {
}
```

Wenn die Katze die Spielwiese betritt, erhält das Feld inside den Wert true und die Methode repaint sorgt für die Auffrischung der Grafik des Applets. Entsprechende Anweisungen stehen in mouseExited.

```
public void mouseEntered(MouseEvent e) {
    inside = true;
    repaint();
}
```

Bei Bewegungen der Katze ermitteln Sie die Koordinaten ihres neuen Ortes. Mit der Methode watch misst die Maus die Entfernung zur Katze.

```
public void mouseMoved(MouseEvent e) {
    tomX = e.getX();
    tomY = e.getY();
    if (!over) {
        watch();
    }
    repaint();
}
```

Bei weniger als 90 Pixel hängt es von einer Zufallszahl ab, ob die Maus wegspringt. Die Klasse Math enthält statische Methoden für Funktionen, wie random zur Ermittlung einer Zufallszahl zwischen 0.0 und 1.0. Der Casting-Operator (int) sorgt dafür, dass die Nachkommastellen einer Fließkommazahl abgeschnitten werden, so dass eine Ganzzahl entsteht.

```
private void watch() {
    if (getDistance() < 90) {
        if (Math.random() < 0.7) {
```

```
while (getDistance() < 95)
{
    jerryX = 32 + (int)
(Math.random() * 237);
    jerryY = 34 + (int)
(Math.random() * 233);
}
}
```

Bei der Berechnung der Entfernung kommt die Wurzelfunktion sqrt zum Einsatz. Mit return geben Sie das Ergebnis zurück.

```
private int getDistance() {
    int dx = jerryX - tomX;
    int dy = jerryY - tomY;
    return (int) Math.sqrt(dx * dx
+ dy * dy);
}
```

Beim Drücken einer Maustaste wird nur reagiert, wenn die Maus noch lebt. Bei einer Entfernung kleiner als 45 ist sie gefangen. Eine Siegesmelodie aus 20 zufällig gewählten Tönen erklingt. Der Ausdruck (int) (d / 0.2) + 1 liefert eine zufällige Zahl zwischen 1 und 5.

```
public void mousePressed(MouseEvent e) {
    if (!over) {
        if (getDistance() < 45) {
            over = true;
            for (int i = 0; i < 20;
i++) {
                double d = Math.random();
                clip[(int) (d / 0.2) +
1].play();
                snooze();
            }
            repaint();
        } else {
```

JUMPINGMOUSE.JAVA

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;
import javax.swing.*;

public class JumpingMouse extends Applet implements MouseListener, MouseMotionListener {

    private boolean inside;

    private boolean over;

    private int jerryX = 150;
    private int jerryY = 150;

    private int tomX;
    private int tomY;

    private Image jerry;
    private Image tom;
    private Image buffer;

    private Graphics screen;
    private AudioClip[] clip;

    public void init() {
        jerry = getImage("Jerry.gif");
        tom = getImage("Tom.gif");
        buffer = createImage(300, 300);
        screen = buffer.getGraphics();
        clip = new AudioClip[] {getAudioClip("Hahahahaha.au"), getAudioClip("1.au"),
            getAudioClip("2.au"), getAudioClip("3.au"), getAudioClip("4.au"),
            getAudioClip("5.au")};
        addMouseListener(this);
        addMouseMotionListener(this);
    }

    private Image getImage(String name) {
        try {
            return super.getImage(getCodeBase(), name);
        } catch (NullPointerException e) {
            return Toolkit.getDefaultToolkit().getImage(name);
        }
    }

    private AudioClip getAudioClip(String name) {
        try {
            return super.getAudioClip(getCodeBase(), name);
        } catch (NullPointerException e) {
            try {
                URL url = new URL("file:" + new File(".").getAbsolutePath() +
                    File.separator + name.substring(0, name.indexOf('.')) + ".wav");
                return Applet.newAudioClip(url);
            } catch (MalformedURLException ex) {
                return null;
            }
        }
    }

    public Dimension getPreferredSize() {
        return new Dimension(300, 300);
    }

    public void update(Graphics g) {
```



JUMPINGMOUSE.JAVA (FORTSETZUNG)

```

    paint(g);
}

public void paint(Graphics g) {
    screen.setColor(new Color(255, 255, 202));
    screen.fillRect(0, 0, 300, 300);
    if (over) {
        screen.setColor(Color.black);
        screen.fillRect(jerryX - 12, jerryY - 34, 10, 70);
        screen.fillRect(jerryX - 32, jerryY - 14, 50, 10);
    } else {
        screen.drawImage(jerry, jerryX - 32, jerryY - 34, this);
    }
    if (inside) {
        screen.drawImage(tom, tomX - 45, tomY - 55, this);
    }
    g.drawImage(buffer, 0, 0, this);
}

public void mouseClicked(MouseEvent e) {
}

public void mouseDragged(MouseEvent e) {
    mouseMoved(e);
}

public void mouseEntered(MouseEvent e) {
    inside = true;
    repaint();
}

public void mouseExited(MouseEvent e) {
    inside = false;
    repaint();
}

public void mouseMoved(MouseEvent e) {
    tomX = e.getX();
    tomY = e.getY();
    if (!over) {
        watch();
    }
    repaint();
}

private void watch() {
    if (getDistance() < 90) {
        if (Math.random() < 0.7) {
            while (getDistance() < 95) {
                jerryX = 32 + (int) (Math.random() * 237);
                jerryY = 34 + (int) (Math.random() * 233);
            }
        }
    }
}

private int getDistance() {
    int dx = jerryX - tomX;
    int dy = jerryY - tomY;
    return (int) Math.sqrt(dx * dx + dy * dy);
}

public void mousePressed(MouseEvent e) {
    if (!over) {
        if (getDistance() < 45) {
            over = true;
            for (int i = 0; i < 20; i++) {
                double d = Math.random();
                clip[(int) (d / 0.2) + 1].play();
                snooze();
            }
            repaint();
        } else {
            clip[0].play();
        }
    }
}

private void snooze() {
    try {
        Thread.sleep(100);
    } catch (InterruptedException e) {
    }
}

public void mouseReleased(MouseEvent e) {
}

public static void main(String[] arguments) {
    JFrame frame = new JFrame("Fang die Maus!");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JumpingMouse applet = new JumpingMouse();
    frame.getContentPane().add(applet);
    frame.pack();
    Toolkit kit = Toolkit.getDefaultToolkit();
    Dimension dim = kit.getScreenSize();
    frame.setLocation((dim.width - frame.getWidth()) / 2,
        (dim.height - frame.getHeight()) / 2);
    frame.show();
    applet.init();
}
}

```

```

        clip[0].play();
    }
}

```

Die Töne liegen auf den Plätzen der Aufstellung clip des Typs AudioClip[]. Sie ist mit einer Aufstellung von Schachfiguren vergleichbar. Die Indexe fangen bei 0 an, so dass Sie über den Index 1 an das linke schwarze Pferd herankommen.

In der Methode init werden die Plätze von clip initialisiert.

```

clip = new AudioClip[] {get
AudioClip("Hahahahaha.au"), get
AudioClip("1.au"),
getAudioClip("2.au"), get
AudioClip("3.au"), getAudioClip
("4.au"),
getAudioClip("5.au")};

```

Bei erfolglosen Fangversuchen ertönt clip[0], so dass die Maus lacht. Zwischen den einzelnen Tönen sorgen Sie für ein Nickerchen von 0.1 Sekunde.

```

private void snooze() {
    try {
        Thread.sleep(100);
    } catch (InterruptedException
e) {
    }
}

```

Wenn es vorab abgebrochen wird, entsteht eine InterruptedException. Der try-catch Mechanismus erlaubt es, auf entstandene Ausnahmen zu reagieren. Wenn Sie den Befehl

```
jar -xvf src.jar
```

zum Entpacken des Archivs src.jar in C:\jdk1.3 ausführen, finden Sie die Klasse Applet im Ordner java\applet in C:\jdk1.3\src. Um sie bekannt zu machen, schreiben Sie die Zeile

```
import java.applet.*;
```

an den Anfang der Klasse JumpingMouse. Das Sternchen sorgt für den Import aller Typen in java\applet, also auch von AudioClip.

Zur Applikation übergehen

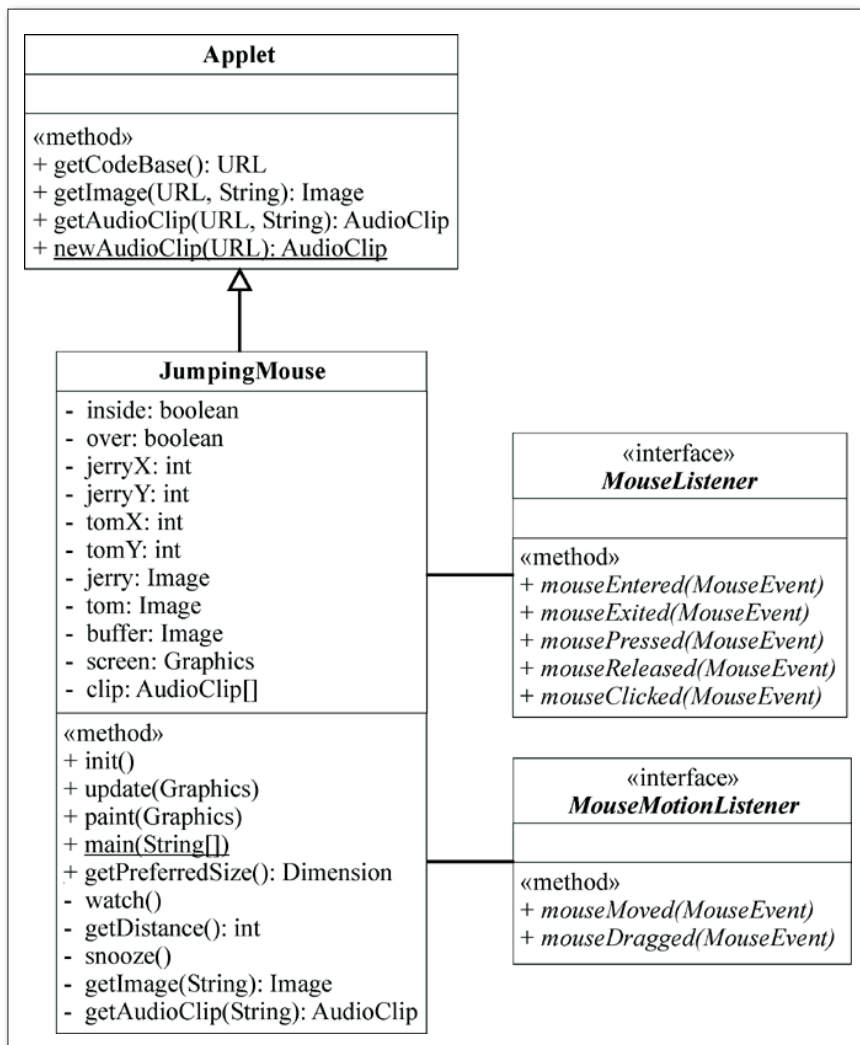
Um JumpingMouse als Applikation zu nutzen, fügen Sie eine statische Hauptmethode ein. Das Applet kommt in ein Fenster, das mit der Methode show auf dem Bildschirm erscheint.

```

public static void main(String[]
arguments) {
    JFrame frame = new JFrame("Fang
die Maus!");
    <weitere Anweisungen>
    frame.show();
}

```

Die Wiesengröße ermittelt das Fenster



UML-DIAGRAMM von JumpingMouse. UML heißt Unified Modeling Language.

auf eigene Faust.

```

public Dimension
getPreferredSize() {
    return new Dimension(300, 300);
}
    
```

In Applikationen sind die speziellen Funktionen von Browsern nicht bekannt. Daher entsteht beim Aufruf der Methode `getCodeBase` zur Ermittlung der URL des Ordners `JumpingMouse` eine `NullPointerException`.

Die üblichen Methoden in der Klasse

Applet zum Laden von Bildern und Klängen scheitern. Mit dem alternativen Ausdruck

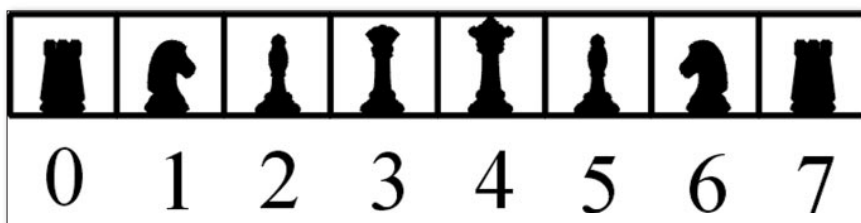
```

Toolkit.getDefaultToolkit().
getImage(name)
    
```

erhalten Sie das Bild in der Datei `name`. Die URL der Sounddateien setzen Sie selbst zusammen.

```

URL url = new URL("file:" + new
File(".").getAbsolutePath() +
File.separator
+ name.substring(0,
name.indexOf('.') + ".wav");
    
```



EINE AUFSTELLUNG mit Schachfiguren.

Die Erweiterung `au` wird durch `wav` ersetzt, so dass der Spieler einen höheren Klanggenuss erfährt. Mit

```
Applet.newAudioClip(url)
```

erhalten Sie den gewünschten Klang.

Java bietet vielfältige weitere Möglichkeiten. In den Tutorials und Büchern im Kasten "Informationsquellen" finden Sie weitere Informationen zu interessanten Themen rund um Java und zu den anderen Klassen der Bibliothek.

Die Tools des Java 2 SDK

Ein Texteditor und eine Eingabeaufforderung reichen aus, um Java-Programme zu schreiben. Der Compiler

```
javac JumpingMouse.java
```

übersetzt den Quellcode in den Bytecode `JumpingMouse.class`. Der Interpreter

```
java JumpingMouse
```

führt das Spiel als Applikation aus. Der Applet Viewer

```
. appletviewer JumpingMouse.html
```

zeigt die Applets auf der Webseite an.

Als Starthilfe stehen die letzten beiden Befehle in den Dateien `Application.bat` und `Applet.bat`. Wenn es Ihnen nicht gelingt, die Maus zu fangen, erhöhen Sie die Zahl 45 in der Methode `mousePressed`.

INFORMATIONSQUELLEN

Java-Tutorial von Sun

<http://java.sun.com/docs/books/tutorial/>

Go To Java 2 von Guido Krüger

<http://www.javabuch.de/>

BÜCHER

James Gosling, Bill Joy, Guy Steele, Gilad Bracha:

The Java Language Specification

Addison-Wesley, 2000, ISBN 0-201-31008-2, US \$ 39.95

Roland Willms:

Java - echt einfach

Franzis, 2001, ISBN 3-7723-7165-5, DM 29.95

Roland Willms:

Java Programmierung Praxisbuch

Franzis, 2000, ISBN 3-7723-7604-5, DM 99.95