Übungsaufgabe 4
Übung IT/EP 188.156

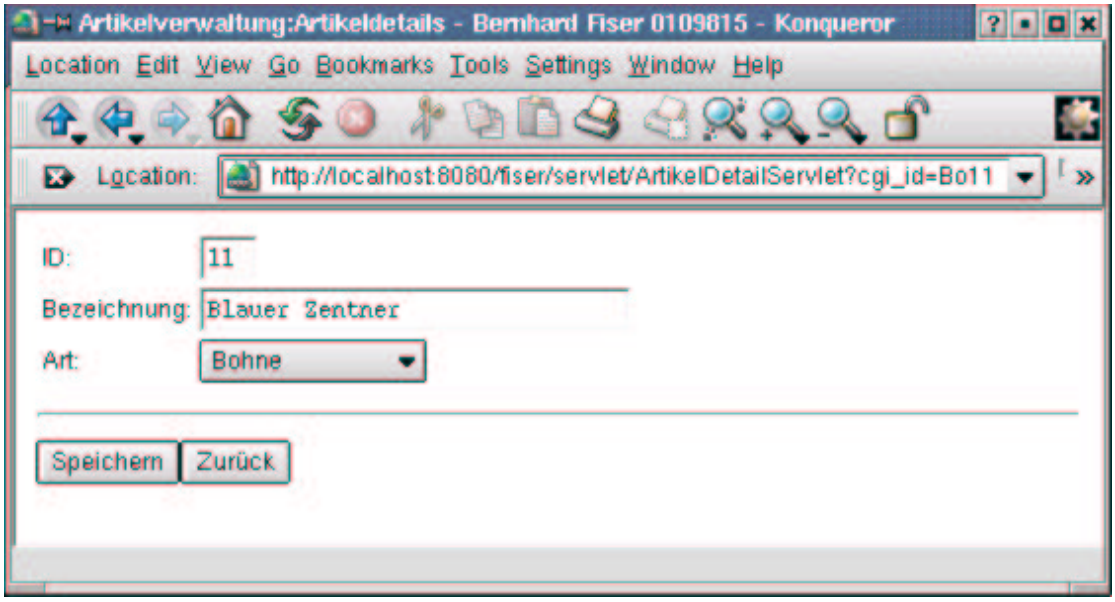
Bernhard Fiser
0109815

Beschreibung
Die Artikelverwaltung dient zur Verwaltung eines Lagers eines Gemüselieferanten. Es können die Artikel erfasst werden, diese bekommen eine Kürzel anhand ihrer Gemüseart (Bohne, Gurke, ...) und müssen mit einer forlaufenden Nummer versehen werden.

Das Problem wurde durch eine Webanwendung gelöst, da dadruch im Unternhemen "DynamoSeeds" mehrere Mitarbeiter gleichzeitig Artikel ändern, löschen oder anlegen können.

Die Hauptmaske wird durch das ArtikelVerwaltungServlet realisiert. Dieses Servlet stellt eine Liste aller im Lager vorhandenen Artikel da, und von dort aus können diese auch bearbeitet, gelöscht oder neue Artikel angelegt werden.

Der Button "Löschen" entfernt einen Artikel aus der Liste, mit "Ändern" oder "Neuen Artikel anlegen" kommt man auf die Detailmaske, welche durch das ArtikelDetailServlet realisiert wurde. Das ArtikelDetailServlet liest die Daten aus der ArtikelVerwaltung und zeigt die Daten des Ausgewählten Objekts anhand der ArtikelID an. Wird beim Ändern eines vorhandenen Artikels die ID verändert, so wird beim Speichern ein neuer Artikel mit dieser ID angelegt, der alte bleibt unberührt. Wird beim geählten Artikel nur die Beschreibung geändert, so wird diese beim Speichern im bestehenden Artikel überschrieben. Durch "Neuen Artikel anlegen" wird diese Maske leer präsentiert, und die Daten werden in einen neuen Artikel beim Speichern übernommen.



Der Kern der Anwendung wird durch die Klasse ArtikelVerwaltung realisiert. Diese Klasse ist so programmiert, daß es nur eine einzige Instanz gibt. Das ist wichtig, da auf die Webanwendung mehrere Personen gleichzeitig zugreifen können, jedoch nur ein einziger Lagerstamm existiert. Das wurde realisiert, in dem der Constructor "private" ist, und eine eigene "static" Methode "getInstance" programmiert wurde, die die Inititalisierung des Objektes übernimmt. Sollte es bereits einmal instanziert worden sein, dann wird die Referenz auf dieses Objektes zurückgegeben. Beim ersten aufruf des Servlets werden die Daten von der Platte gelesen. Das lesen der Daten kann auch lokal von der Shell getestet werden, um mögliche Syntaxfehler in den Datenfiles zu überprüfen. Zu diesem Zweck wurde die Methode "main" in der Klasse ArtikelVerwaltung realisiert. Die ArtikelVerwaltung selbst ist eine erweiterte HashMap, da dort die einzelnen Artikel Objekte abgelegt werden.

Als weitere Klassen existiert die Klasse Art und die Klasse Artikel. Die Klasse Art dient zum Speichern der einzelnen Gemüsearten, diese wiederum werden innerhalb der ArtikelVerwaltung als HashMap abgelegt. Die Klasse Artikel dient zum Speichern der Daten eines einzlnen Artikels, diese wiederum werden in der ArtikelVerwaltung selbst abgelegt.

Pseudocode

- START (duch aufruf des ArtikelVerwaltungServlet)
- Instanz der ArtikelVerwaltung holen:
- − Falls schon eine Instanz existiert, diese zurückgeben.
- − Instanz der ArtikelVerwaltung erzeugen.
- − Andernfalls Artendatei öffnen.
- − Iteration über alle Zeilen.
- − Artenobjekt erstellen.
- − Zeile parsen und Werte in Objekt übernehmen.
- − Falls Fehler beim parsen −> EXIT.
- − Artenobjekt in Hashmap aufnehmen.
- − Artendatei schließen.
- − Artikeldatei öffnen.
- − Iteration über alle Zeilen.
- − Artikelobjekt erstellen.
- − Zeile parsen und Werte in Objekt übernehmen.
- − Falls Fehler beim parsen −> EXIT.
- − ArtikelObjekt in HashMap (ArtikelVerwaltung) aufnehemen.
- − Artikeldatei schließen.
- − Instanz der ArtikelVerwaltung zurückgeben.
- HTML−Seite beginnen.
- Prüfen ob Parameter "cgi_save" (Speichern) übergeben.
- Falls ja
- neues Artikelobjekt erstellen
- Parameter für ID, Beschreibung und Art in Objekt übernehmen
- Objekt in ArtikelVerwaltung aufnehmen
- Artikeldatei speichern.
- Prüfen ob Parameter "cgi_del" (Löschen) übergeben.
- Falls ja
- ID übernehmen und Objekt anhande der ID in Artikelverwaltung suchen.
- Objekt löschen.
- Artikeldatei speichern.
- Liste sortieren mithilfe eines abgeänderten Selection−Sort:
- − neue LinkedList erstellen.
- − Iteration über alle Objekte der ArtikelVerwaltung.
- − Artikelobjekt auslesen.
- − Positionszähler auf 0 setzen.
- − Iteration über alle Objekte der LinkedList.
- − Objekt aus LinkedList lesen
- − Vergleiche ID des Objektes aus ArtikelVerwaltung mit ID
- des Objektes aus LinkedList
- − Falls ID größer, innere Schleife beenden.
- − Positionszähler erhöhen.
- − Das Objekt an der Stelle des Positionszählers in LinkedList
- aufnehmen.
- − LinkedList zurückliefern.
- Iteration über alle Objekte der LinkedList.
- Objekt aus der Liste holen.
- Objekt auf Bildschirm ausgeben.
- HTML−Seite beenden.
- ENDE

Sourcecode Art.java

```
import java.util.*;

/** Klasse zur Verwaltung der Gemüsearten.
  *
  * @author Bernhard Fiser 0109815
  * @version 1.0
  */
public class Art
{
	private String id;
	private String desc;

	/** Erstellt eine neue Leere Art */
	public Art()
	{
		this.id = "";
		this.desc = "";
	}

	/** Liefert die ID der Art zurück
	  * @return String
	  */
	public String getID()
	{
		return(this.id);
	}

	/** Liefert die textuelle Beschreibung der Art zurück.
	  * @return String
	  */
	public String getDescription()
	{
		return(this.desc);
	}

	/** Parsed eine Textzeile und übernimmt die Daten in das Objekt.
	  * @param String : s
	  * @return boolean : true ... Parser erfolgreich
	  * @return boolean : false .. Syntax error
	  */
	public boolean parseLine(String s)
	{
		String		t;
		StringTokenizer		st;
		int		i;

		/* Stringtokenizer und Zählvariable initialisieren */
		i = 0;
		st = new StringTokenizer(s, " ");
		while (st.hasMoreTokens())
		{
			/* Token aus string lesen */
			t = st.nextToken();
			switch (i)
			{
				/* erster Token in ID übernehmen */
				case 0 :
					/* Falls strlen ungleich 2, dann Syntaxerror */
					if (t.length() != 2)
						return(false);
					this.id = t;
					break;

				/* zweiter Token in Beschreibung übernehmen */
				case 1 :
					this.desc = t;
					break;

				/* weitere Tokens nicht erlaubt -> Syntaxerror */
				default :
					return(false);
			}
			/* Tokenzähler erhöhen */
			i++;
		}
		/* Falls weniger als 2 Tokens -> Syntaxerror */
		if (i < 2)
			return(false);
		/* Andernfalls alles OK */
		return(true);
	}

}
```

Sourcecode Artikel.java

```
import java.util.*;

/**
 * Klasse zur Speicherung eines einzelnen Artikelobjektes
 *
 * @author Bernhard Fiser 0109815
 * @version 1.1
 */
public class Artikel
{
	private String desc;
	private Art    art;
	private int    artn;

	/**
	 * Constructor
	 * Legt ein neues Objekt an und initialisiert die internen
	 * privaten Variablen.
	 * @param Art : initielles Artobjekt.
	 */
	public Artikel(Art art)
	{
		this.art = art;
		this.desc = "";
		this.artn = 0;
	}

	/** Liefert die ID des Artikels zurück */
	public String getID()
	{
		if (this.art == null)
			return(null);
		return(this.art.getID() + this.artn);
	}

	/** Liefert numerischen Teil der ID des Artikels zurück */
	public int getIDn()
	{
		return(this.artn);
	}

	/** liefert die Art zurück
	  * @return Art art : Art */
	public Art getArt()
	{
		return(this.art);
	}

	/** liefert die Beschreibung des Artikels zurück
	 * @return String desc : Beschreibung */
	public String getDescription()
	{
		return(this.desc);
	}

	/** Parsed string und belegt die internen Instanzvariablen
	  * @param String :   String der Parameter enthält
	  * @param HashMap :  HashMap der Art-Objekte
	  * @return boolean : true ... alles OK
	  * @return boolean : false .. Syntax error
	  */
	public boolean parseLine(String s, HashMap h)
	{
		String         t,
		               a = "",
		               b = "";
		StringTokenizer       st;
		int           i;

		/* Stringtokenizer initialisieren */
		i = 0;
		st = new StringTokenizer(s, ";");
		while (st.hasMoreTokens())
		{
			/* Token lesen */
			t = st.nextToken();
			switch (i)
			{
				/* ersten Token in Variable übernehmen */
				case 0 :
					b = t;
					break;

				/* Zweiten Token in Beschreibung des Artikels übernehmen
*/
				case 1 :
					this.desc = t;
```

```
                                        break;

                                /* Dritten Token in numerische ID-Variable übernehmen */
                                case 2 :
                                        /* Falls strlen weniger als 3 -> Syntaxerror */
                                        if (t.length() < 3)
                                                return(false);
                                        /* Numerischen Anteil in int konvertieren */
                                        a = t.substring(0, 2);
                                        try
                                        {
                                                this.artn                                       =
Integer.parseInt(t.substring(2));
                                        }
                                        /*   Falls   Konvertierung   nicht   möglich   ->
Syntaxerror */
                                        catch (Exception e)
                                        {
                                                return(false);
                                        }
                                        break;

                                /* Weitere Tokens -> Syntaxerror */
                                default :
                                        return(false);
                        }
                        /* Tokenzähler erhöhen */
                        i++;
                }
                /* Falls weniger als 3 Tokens -> Syntaxerror */
                if (i < 3)
                        return(false);

                /* Falls Art nicht in HashMap gefunden -> Syntaxerror */
                if ((this.art = (Art) h.get(a.toLowerCase())) == null)
                        return(false);

                /*  Falls  Beschreibung  der  Art  nicht  zu  erstem  Token  passt  ->
Syntaxerror */
                if (b.compareToIgnoreCase(this.art.getDescription()) != 0)
                        return(false);

                /* Andernfalls alles OK */
                return(true);
        }
}
```

Sourcecode ArtikelDetailServlet.java

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;


/** Servlet Eingabe eines einzelnen Artikels
  * @author Bernhard Fiser 0109815
  */
public class ArtikelDetailServlet extends HttpServlet
{
	private static ArtikelVerwaltung
				av;
	private Artikel		a;


	/** Beim Laden des Servlets einmal instanzieren */
	public void init()
	{
		av = ArtikelVerwaltung.getInstance();
	}

	/** CGI-POST Methode, wird auf die CGI-GET Methode weitergeleitet */
	public void doPost(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException
	{
		this.doGet(request, response);
	}

	/** CGI-GET Methode, hier findet gesamte Verarbeitung statt */
	public void doGet(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException
	{
		Art		art;
		Iterator	i;
		String		id,
				selected;

		/* Contenttype der HTML-Seite setzen */
		response.setContentType("text/html");
		/* outputhandle erstellen */
		PrintWriter out = response.getWriter();
		/* HTML Seite beginnen */
		out.println(
			"<html><head><meta name=\"author\" content=\"Bernhard Fiser\">"
+
			"<meta name=\"date\" content=\"2002-05-14T08:29:56+02:00\">" +
			"<title>Artikelverwaltung:Artikeldetails - Bernhard Fiser
0109815</title></head>" +
			"<body bgcolor=\"#ffffff\">\n<form method=\"post\"
action=\"ArtikelVerwaltungServlet\">");

		/* CGI-Parameter 'cgi_id' aus FORM übernehmen... */
		id = request.getParameter("cgi_id");
		/* ... und falls dieser vorhanden .. */
		if ((id == null) || (id.length() < 3))
			id = "";

		/* Artikel aus Datei lesen */
		this.init();
		/* neue Artikel erzeugen, falls keine sinnvolle ID vorhanden ist */
		if ((a = (Artikel) av.get(id.toLowerCase())) == null)
			a = new Artikel((Art) av.getArten().get("bo"));

		/* Tabelle mit Inputfelder erstellen, und defaultwerte Eintragen */
		out.println("<table border=\"0\">");
		out.println("<tr><td>ID:</td><td><input type=\"text\" name=\"cgi_id\"
value=\"" + a.getIDn() + "\" size=\"3\"></td>");
		out.println("<tr><td>Bezeichnung:</td><td><input type=\"text\"
name=\"cgi_desc\" value=\"" + a.getDescription() + "\" size=\"30\"></td>");
		out.println("<tr><td>Art:</td><td><select name=\"cgi_art\">");

		/* Iteration über alle Arten um ein popup (select) zu erstellen */
		i = av.getArten().values().iterator();
		while (i.hasNext())
		{
			art = (Art) i.next();

			/* falls der Artikel zu dieser Art gehört, dann diesen Parameter
vom select auswählen */
			if
(art.getDescription().compareToIgnoreCase(a.getArt().getDescription()) == 0)
				selected = " selected";
			else
				selected = "";

			out.print("<option value=\"" + art.getID() + "\"" + selected +
```

```
">" + art.getDescription());
                }

                out.println("</select></td>");
                out.println("</table><hr><input    type=\"submit\"    name=\"cgi_save\"
value=\"Speichern\">");
                out.println("<input          type=\"submit\"          name=\"cgi_list\"
value=\"Zur&uuml;ck\">");
                out.println("</body></html>\n");
        }
}
```

Sourcecode ArtikelVerwaltung.java

```
import java.util.*;
import java.io.*;


/** Klasse zur Verwaltung der Artikel
  * @author Bernhard Fiser 0109815
  * @version 1.0 */
public class ArtikelVerwaltung extends HashMap
{
	private static ArtikelVerwaltung	hmap = null;
	private static HashMap			arten;
	private String				path				=
"/home/fiser/allgemein/studium/itep/abgabe4/testdaten";

	/** Constructor
	  * legt eine neue Hashmap an, in der die Artikelobjekte
	  * gespeichert werden. Der Constructor ist privat und kann daher nicht
	  * von 'außerhalb' instanziert werden. */
	private ArtikelVerwaltung()
	{
		super();
		arten = new HashMap();
	}

	/** Pseudo-Constructor zum instanzieren eines Objekts.
	  * Die Instanzierung erfolgt nur, falls noch kein Objekt dieser Klasse
	  * existiert, andernfalls wird dieses Objekt einmal erstellt. */
	public synchronized static ArtikelVerwaltung getInstance()
	{
		String		line;
		ReadFile	f;

		/* Falls Artikelverwaltung besteht, diese zurückgeben */
		if (hmap != null)
			return(hmap);

		/* Neue Artikelverwaltung erstellen */
		hmap = new ArtikelVerwaltung();

		/* Artenfile einlesen */
		if (!hmap.readArten())
		{
			System.err.println("*** error: readArten");
			hmap = null;
			arten = null;
			return(null);
		}

		/* Artikelfile einlesen */
		if (!hmap.readArtikel())
		{
			System.err.println("*** error: readArtikel");
			hmap = null;
			arten = null;
			return(null);
		}

		/* Artikelverwaltung zurückgeben */
		return(hmap);
	}

	/** Methode zum lesen der Arten
	  * @return true ... alles OK
	  * @return false .. Syntaxerror
	  */
	public synchronized boolean readArten()
	{
		int		lineno = 0;
		Art		a;
		String		line;
		ReadFile	f;

		/* File öffnen */
		f = new ReadFile(path, "arten.txt");
		/* Zeilenweise lesen */
		while ((line = f.readLine()) != null)
		{
			lineno++;
			/* Neues Artenobjekt erstellen */
			a = new Art();
			/* Zeile parsen */
			if (!a.parseLine(line))
			{
				/* Bei Syntaxerror Fehler ausgeben */
				System.err.println("*** error: syntax error in line " +
lineno);
				return(false);
```

```
                        }
                        /* Artenobjekt in ArtenHashmap aufnehmen */
                        this.arten.put(a.getID().toLowerCase(), a);
                }
                /* Datei schließen */
                f.close();
                /* Alles OK -> true zurückgeben */
                return(true);
        }

        /** Methode zum Lesen der Artikel
          * @return true ... alles OK
          * @return false .. Syntaxerror
          */
        public synchronized boolean readArtikel()
        {
                int             lineno = 0;
                Artikel                 a;
                String          line;
                ReadFile        f;

                /* Datei öffnen */
                f = new ReadFile(path, "artikel.txt");
                /* Zeilenweise lesen */
                while ((line = f.readLine()) != null)
                {
                        lineno++;
                        /* Neuse Artikelobjekt erstellen */
                        a = new Artikel((Art) arten.get("bo"));
                        /* Zeile parsen */
                        if (!a.parseLine(line, this.arten))
                        {
                                /* Bei Syntaxerror -> Fehlerausgeben */
                                System.err.println("*** error: syntax error in line " +
lineno);
                                return(false);
                        }
                        /* Artikel in Artikelverwaltung aufnehmen */
                        this.put(a.getID().toLowerCase(), a);
                }
                /* Datei schließen */
                f.close();
                /* Alles OK -> true zurückgeben */
                return(true);
        }

        /** Methode zum speichern der Artikelveraltung
          * @return true : falls Speichern erfolgreich
          * @return false : falls Fehler aufgetreten ist */
        public synchronized boolean save()
        {
                Artikel                 a;
                Iterator        i;
                PrintWriter     out;

                /* Falls Hashmap nicht initialisiert, Fehler zurückgeben
                 * (sollte niemals eintreten)*/
                if (hmap == null)
                        return(false);

                /* Datei zum Schreiben öffnen */
                try
                {
                        out  =  new  PrintWriter(new  FileOutputStream(new  File(path  +
"/artikel.txt")));
                }
                /* Bei Fehler -> false zurückgeben */
                catch (IOException e)
                {
                        return(false);
                }

                /* Über alle Objekte der Artikelverwaltung iterieren */
                i = this.values().iterator();
                while (i.hasNext())
                {
                        /* Artikel aus Artikelverwlatung auslesen ... */
                        a = (Artikel) i.next();
                        /* ... und in Datei schreiben */
                        out.println(a.getArt().getDescription()      +      ";"      +
a.getDescription() + ";" + a.getID());
                }
                /* Datei schließen */
                out.close();

                /* Alles OK -> true zurückgeben */
                return(true);
        }

        /** Arten zurückliefern
```

```
          * @return HashMap */
        public synchronized HashMap getArten()
        {
                return(this.arten);
        }

        /** Hauptprogramm zum Testen der Dateien, da
          * ein derartiger Test im Servlet nicht (nur umständlich)
          * möglich ist. */
        public static void main(String argv[])
        {
                ArtikelVerwaltung     av;

                if ((av = ArtikelVerwaltung.getInstance()) == null)
                {
                        System.err.println("*** error: could not get instance");
                        System.exit(1);
                }
                System.out.println(av.size() + " Artikel geladen");
        }
}
```

Sourcecode ArtikelVerwaltungServlet.java

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;


/** Hauptservlet der Artikelveraltung
  * Hierin wird die Artikelliste erzeugt, die Artikel gelöscht und gespeichert
  *
  * @author Bernhard Fiser 0109815
  * @version 1.1
  */
public class ArtikelVerwaltungServlet extends HttpServlet
{
	private static ArtikelVerwaltung
			av;

	/** Servletinitialisierung beim erstmaligen Laden des Servlets
	   * vom Tomcat */
	public void init()
	{
		av = ArtikelVerwaltung.getInstance();
	}

	/** Methode für CGI-POST. Diese wird auf die doGet Methode
	  * weitergeleitet */
	public void doPost(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException
	{
		this.doGet(request, response);
	}

	/** Methode für CGI-GET. Hier findet die Verarbeitung der Daten statt */
	public void doGet(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException
	{
		LinkedList
			l;
		int	c,
			idn;
		String	id,
			bg,
			desc;
		Art	art;

		/* Contenttype für output setzen */
		response.setContentType("text/html");
		/* Outputstream festhalten */
		PrintWriter out = response.getWriter();
		/* HTML-Doc Einleitung */
		out.println(
			"<html><head><meta name=\"author\" content=\"Bernhard Fiser\">"
+
			"<meta name=\"date\" content=\"2002-05-14T08:28:00+02:00\">" +
			"<title>Artikelverwalung V1.0 - Bernhard Fiser
0109815</title></head><body>\n");

		/* Falls der CGI-Parameter 'cgi_save' übergeben wurde,
		 * Artikel in Artikelverwaltung aufnehmen und speichern */
		if (request.getParameter("cgi_save") != null)
		{
			id = request.getParameter("cgi_art");
			art = (Art) av.getArten().get(id.toLowerCase());
			try
			{
				/* CGI-Parameter aus FORM übernehmen und convertiertn */
				idn = Integer.parseInt(request.getParameter("cgi_id"));
			}
			catch (NumberFormatException e)
			{
				/* Falls Konvertierung fehlgeschlagen, Werte auf 0
setzen */
				idn = 0;
			}
			/* restliche CGI-Parameter übernehmen. */
			desc = request.getParameter("cgi_desc");

			/* int Werte auf Plausibilität prüfen */
			if ((idn > 0) && (art != null))
			{
				/* Falls Werte in Ordnung, neuen Artikel anlegen... */
				Artikel a = new Artikel((Art) av.getArten().get("bo"));
				if (a.parseLine(art.getDescription() + ";" + desc + ","
+ art.getID() + idn, av.getArten()))
				{
					/* ... und in Artikelverwaltun aufnehmen... */
					av.remove(a.getID().toLowerCase());
```

```
                                        av.put(a.getID().toLowerCase(), a);
                                        /* ... und speichern */
                                        if (!av.save())
                                                out.println("<font
color=\"red\"><b>Fataler Fehler beim Speichern!</b></font><br>");
                                        else
                                                /* Falls Speicherung erfolgreich, dann
OK-Meldung ausgeben */
                                                out.println("<font
color=\"green\">Artikel Nummer " + a.getID() + " wurde erfolgreich
gespeichert!</font><br>");
                                }
                                else
                                        /* Fataler Fehler (dürfte eigentlich nicht
vorkommen) */
                                        out.println("<font color=\"red\">*** fatal:
Artikel konnte nicht angelegt werden! Eingabedaten sind falsch.</font><br>");
                        }
                        else
                                /* Falls Plausibilitätsprüfung fehlgeschlagen, Fehler
ausgeben */
                                out.println("<font color=\"red\">Artikel konnte nicht
gespeichert werden! Eingabedaten sind falsch.</font><br>");
                }

                /* Falls CGI-Parameter 'cgi_del' übergeben wurde, Artikelobjekt
                 * aus der Artikelverwaltung löschen */
                if (request.getParameter("cgi_del") != null)
                {
                        /* Übergebene Artikelnummer konvertieren und auf
                         * Plausibilität prüfen */
                        id = request.getParameter("cgi_id");
                        /* Falls Plausibilität OK, Artikel löschen */
                        if (av.remove(id.toLowerCase()) != null)
                        {
                                /* ArtikelVerwaltung speichern */
                                if (!av.save())
                                        out.println("<font color=\"red\"><b>Fataler
Fehler beim Speichern!</b></font><br>");
                                else
                                        /* OK-Meldung ausgeben, falls Speicherung
erfolgreich */
                                        out.println("<font color=\"green\">Artikel Nummer
" + id + " wurde erfolgreich gel&ouml;scht.</font><br>");
                        }
                        else
                                /* Falls Plausibilitätsprüfung fehlgeschlagen, Fehler
ausgeben */
                                out.println("<font color=\"red\"Artikel Nummer " + id +
" konnte nicht gel&ouml;scht werden. Artikel existiert nicht.</font></br>");
                }

                /* Button für Neuanlage generiern*/
                out.println("<form method=\"post\"
action=\"ArtikelDetailServlet\"><input type=\"submit\" value=\"Neuen Artikel
anlegen\"></form><hr>");
                /* Tabelle der vorhandenen Artikel generieren */
                out.println("<table
border=\"1\"><tr><th>ID</th><th>Beschreibung</th><th>Art</th></tr>");
                /* Iterator der Hashmap erstellen */
                l = sortArtikelVerwaltung();
                ListIterator it = l.listIterator();
                c = 0;
                /* Über jedes Objekt iterieren */
                while (it.hasNext())
                {
                        /* Hintergrundfarbe der Tabelle festlegen */
                        if ((c++ % 2) == 0)
                                bg = "#aaeeaa";
                        else
                                bg = "#eeeeaa";
                        /* Artikel auslesen aus der Hashmap */
                        Artikel a = (Artikel) it.next();
                        /* Zeile mit den Artikelattribute ausgeben */
                        out.println(
                                "<tr bgcolor=\"" + bg + "\">" +
                                "<td>" + a.getID() + "</td>" +
                                "<td>" + a.getDescription() + "</td>" +
                                "<td>" + a.getArt().getDescription() + "</td>" +
                                "<td><a href=\"ArtikelDetailServlet?cgi_id=" + a.getID()
+ "\">&Auml;ndern</a></td>" +
                                "<td><a href=\"ArtikelVerwaltungServlet?cgi_id="
a.getID() + "&cgi_del=1\">L&ouml;schen</a></td>" +
                                "</tr>\n");
                }
                /* Tabelle und HTML-Doc beenden */
                out.println("</table></body></html>\n");
        }

        /** Sortierung der Artikelverwaltung für die Darstellung am Bildschirm */
```

```
        private LinkedList sortArtikelVerwaltung()
        {
                Artikel                 a, b;
                LinkedList      l;
                ListIterator    j;
                Iterator        k;
                int             i;

                /* neue LinkedList erstellen und Iterator für Hashmap erstellen */
                l = new LinkedList();
                k = av.values().iterator();
                /* Iteration über alle Elemente der Artikelverwaltung */
                while (k.hasNext())
                {
                        /* Artikel aus Hashmap lesen */
                        a = (Artikel) k.next();
                        i = 0;
                        /* Iteration über Linkedlist */
                        j = l.listIterator(0);
                        while (j.hasNext())
                        {
                                /* Artikel aus LinkedList lesen */
                                b = (Artikel) j.next();
                                /*  Falls  Artikel  der  Liste  größere  Artikelnummer  hat,
als
                                 * Artikel aus Hashmap */
                                if (b.getID().compareToIgnoreCase(a.getID()) > 0)
                                        /* ... dann Schleife abbrechen */
                                        break;
                                i++;
                        }
                        /* Artikel aus Hashmap an der Stelle in die LinkedList
                         * einsortieren, an der die Schleife abgebrochen wurde (s.o.) */
                        l.add(i, a);
                }
                /* LinkedList zurückgehen */
                return(l);
        }
}
```

Sourcecode Readfile.java

```
import java.io.*;

public class ReadFile
{
	private BufferedReader		in;
	private boolean			fehler;

	public ReadFile(String path, String name) {
		try {
			in = new BufferedReader(new FileReader(new File(path, name)));
			fehler = false;
			}
		catch (IOException e) {
			fehler = true;
			System.out.println("Fehler beim Öffnen: " + e);
			}
		}

	public String readLine() {
		String s = null;
		if (fehler)
			return null;
		try {
			s = in.readLine();
			}
		catch (IOException e) {
			System.out.println("Fehler beim Lesen: " + e);
			}
		return s;
		}

	public void close() {
		try {
			fehler = true;
			in.close();
			}
		catch (IOException e) {
			System.out.println("Fehler beim Schließen: " + e);
			}
		}

	/* Testprogramm */
	public static void main(String args[]) {
		String path = "/anydir/ReadFile";
		ReadFile file = new ReadFile(path, "test.txt");

		String s;
		while (null != (s = file.readLine()))
			System.out.println(s);

		file.close();
		}
	}
```