Mit JFreechart Diagramme zeichnen

Malen nach Zahlen

Mit zunehmendem Datenwust gehen für den Betrachter relevante Informationen unter. Sinnvolle Grafiken bringen Wichtiges wieder ans Licht. Die leistungsfähige Bibliothek JFreechart hilft dabei, mit Java aussagefähige Diagramme zu zeichnen und sie in eigene Programme einzubauen. Bernhard Bablok

Ob Business-Daten, Messwerte wissenschaftlicher Untersuchungen oder Zugriffe auf Server: Nichts ist langweiliger als Zahlenkolonnen. Grafiken bringen nicht nur Abwechslung, sondern heben wichtige Werte hervor. Durch die Vielfalt an Anwendungen sind die Anforderungen an entsprechende Pakete aber hoch. Standardsoftware wie Open Office bietet zumindest Unterstützung für die meistbenötigten Charts (siehe Abbildung 1). Doch wer die Daten nicht sowieso schon in einer Tabellenkalkulation hält, muss sie erst einmal in Open Office importieren, selbst wenn der benötigte Charttyp dabei ist. Da ist es oft einfacher, die Grafiken mit einem selbst geschriebenen Programm zu zeichnen.

Die Bibliothek JFreechart [1] der Firma Object-Refinery [2] bietet zwei Möglichkeiten: ein Diagramm interaktiv anfertigen oder es von Hand programmieren. JFreechart ist sehr mächtig und steht unter der LGPL. Kein Wunder also, dass sich die Bibliothek großer Beliebtheit erfreut, wie die lange Liste auf der JFreechart-Homepage beweist.

JFreechart lässt sich auch ohne Handbuch nutzen, dafür sorgen die vielen mitgelieferten Beispiele. Wer mehr Dokumentation braucht, greift zum Developer's Guide, der allerdings fast 40 Dollar kostet. Eine Site-Lizenz schlägt mit knapp 300 Dollar zu Buche. Das gut aufgebaute, fast 350-seitige Handbuch voller Beispiele lohnt sich aber für jeden ernsthaften Nutzer von JFreechart, da es doch sehr mühsam ist, sich alles über die Beispiele und den Quellcode zu erarbeiten. Da der Preis nicht über den eines guten Fachbuchs hinausgeht, ist er auch angemessen.

Download und Installation

Das JFreechart-Paket im Tgz-Format ist 2,3 MByte groß und umfasst den Quellcode, die fertige Jar-Datei »jfreechart-0.9.16.jar« und alle zusätzlich benötigten Bibliotheken. Am einfachsten entpackt man das Archiv etwa unter »/usr/local« mit:

```
tar -xzf jfreechart-0.9.16.tar.gz
 -C /usr/local
```

Eine weiter gehende Installation ist nicht notwendig. Es genügt, wenn sich die Jar-Bibliotheken zur Compile- und Laufzeit im »CLASSPATH« befinden. Im Falle von JFreechart sind dies die Bibliothek »jfreechart-0.9.16.jar« sowie die Bibliotheken im »lib«-Verzeichnis. Die Dokumentation ist wie oben beschrieben nicht dabei. Mittels

```
ant -f ant/build.xml javadoc
```

kann (und sollte) man sich aber die Javadoc-Dokumentation erzeugen. Sie ist eine unverzichtbare Referenz beim Programmieren und hilft dabei, sich in die Klassenstruktur einzuarbeiten.

Einen ersten Überblick über JFreechart geben die Demonstrationsprogramme, die ein Aufruf des Jar-File startet:

```
cd /usr/local/jfreechart-0.9.16
java -jar jfreechart-0.9.16-demo.jar
```

Wie in den Abbildungen 2 und 3 zu sehen, ist das Beispiel nicht durchgängig lokalisiert. Davon abgesehen verschafft es einen guten Eindruck von der Mächtigkeit des Pakets. Ein Klick mit der rechten Maustaste führt zu einem Kontextmenü. Damit lassen sich die Eigenschaften des Diagramms ändern und die Grafik als PNG-Datei speichern. Der übernächste Abschnitt zeigt, wie einfach sich solche Anwendungen mit JFreechart programmieren lassen.

Die zentrale Klasse der Bibliothek ist die Klasse »org.jfree.chart.JFreeChart«. Sie ist ein Container für Titel und Legenden sowie Objekte vom Typ »org.jfree.chart.

plot.Plot« und »org.jfree.data.Dataset«. Plot ist eine abstrakte Klasse, die Subklassen wie »XYPlot« oder »PiePlot« implementieren die eigentliche Funktionalität. Dataset ist eine Schnittstelle (Interface), die von einer Vielzahl von Klassen implementiert wird. Ein Dataset besteht meist aus mehreren Datenserien, beispielsweise setzt sich »XYDataset« aus Objekten vom Typ »XYSeries« zusammen. Eine solche Datenserie umfasst einen Namen und eine Menge von Punkten (x, y). Für Kuchendiagramme (auch Tortendiagramme oder Kreisdiagramme genannt) gibt es ein »DefaultPieDataset«, für Balkendiagramme ein »CategoryDataset« und so weiter. Hier muss der Programmierer ein geeignetes Dataset für seine Anwendung finden.

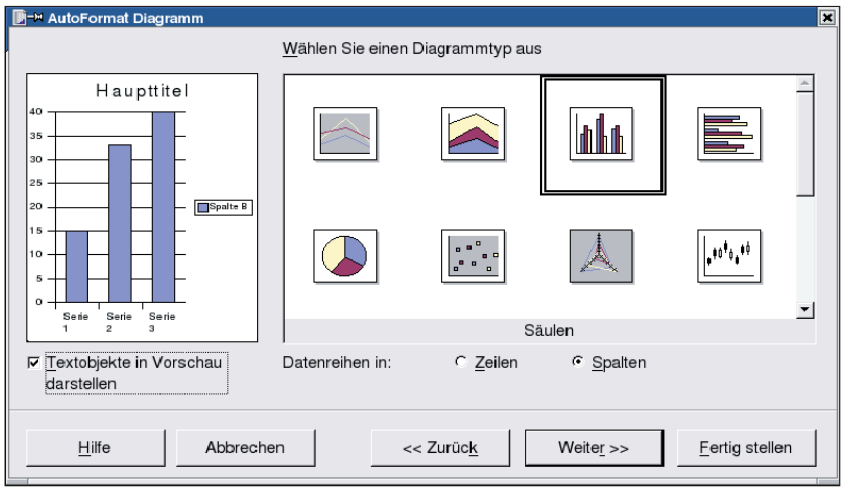

Abbildung 1: Open Office bietet die gängigsten Diagrammtypen wie Säulen-, Balken- und Tortendiagramme.

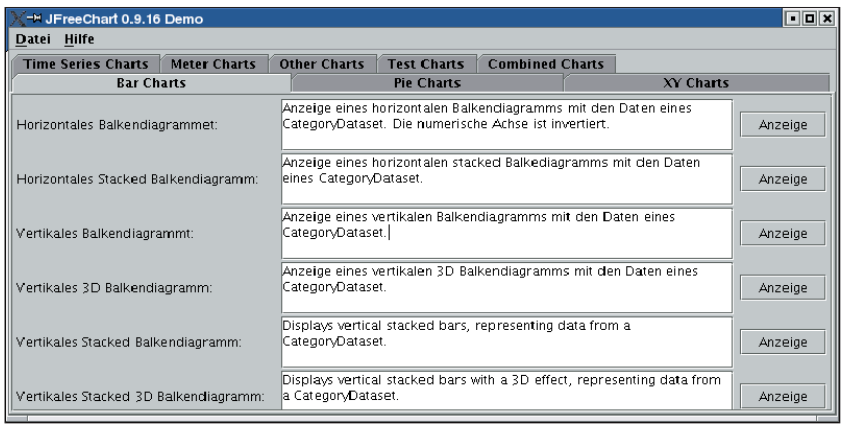

Abbildung 2: Die JFreechart-Beispielanwendung zeigt die Vielfalt an Diagrammen des Pakets. Allein sechs verschiedene Balkendiagramme sind dabei.

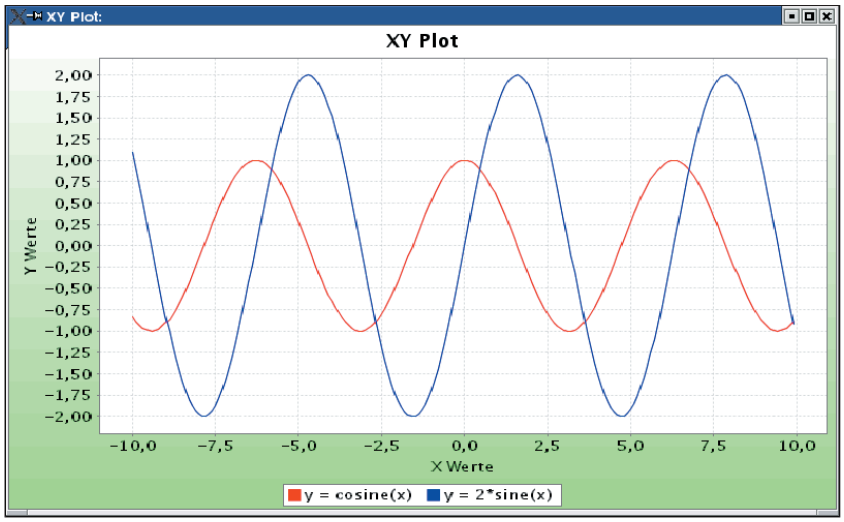

Abbildung 3: Ein »XYPlot« der JFreechart-Demo stellt den Kurvenverlauf von Sinus- und Cosinusfunktionen dar.

Fabrik für Diagramme

Es ist jedoch einfacher, Plot-Objekte zu verwenden. In aller Regel kommt eine statische Methode von »ChartFactory« zum Einsatz, um ein vorkonfiguriertes Diagramm zu zeichnen, das selbst aus einem Dataset erzeugt wurde:

```
JFreeChart chart = ChartFactory.
createPieChart(...);
Plot plot = chart.getPlot();
```

Über das Plot-Objekt lassen sich einige Aspekte des Plots manipulieren, etwa mit der Methode »setBackgroundImage()« ein Hintergrundbild setzen.
Zwei weitere GUI-Klassen sind sehr nützlich: »ChartPanel« und »ChartFrame«. Letztere ist eine einfache Subklasse von JFrame mit einem »ChartPanel« als »ContentPane«. Ein »ChartPanel« stellt all jene Operationen zur Verfügung, die für eine interaktive Bearbeitung des Diagramms wichtig sind: drucken, speichern als PNG, zoomen, bearbeiten der Eigenschaften über ein Menü und so weiter.

Grafik interaktiv

Mit dem Wissen um diese Klassen ist der Weg zum eigenen Diagramm nicht weit und vollzieht sich in vier Schritten: Zuerst erzeugt man ein geeignetes Dataset (Listing 1, Zeilen 47 bis 60), dann ein »JFreeChart« (Zeilen 80 bis 91) über die »ChartFactory«. Fehlt nur noch – zumindest für Standalone-Charts – ein »ChartFrame« für die Anzeige. Soll der Chart in eine eigene Anwendung integriert werden, bietet sich das »ChartPanel« an. Das Diagramm zeigt den freien Platz verschiedener Dateisysteme auf einem USB-Stick und einer DVD-RAM (beide frisch formatiert) in Prozent an. Grundlage ist die Ausgabe von »df -k« in Relation zur Sollkapazität.
Auch wenn es nicht Thema des Coffee-Shops ist: Obwohl auf einem typischen

Backup-Medium wie einer DVD-RAM Journaling nicht nötig ist, lohnt sich sein Einsatz dennoch – ungefähr 250 MByte mehr Platz bringen hier ReiserFS oder JFS gegenüber Ext 2.

Am Ende soll ein Balkendiagramm rauskommen, für das ein »CategoryDataset« erforderlich ist. Die Methode »createDataset()« in den Zeilen 47 bis 60 erzeugt das Dataset. Die Werte sind der freie Platz in Prozent, die Kategorien sind der Medien-Typ (USB-Stick und DVD-RAM) sowie das Dateisystem.

Datenbasis mit XML

Im wirklichen Leben wird niemand die Werte im Programm hart verdrahten. Die Methode »createDatasetFromXML()« (Listing 1, Zeilen 68 bis 72) zeigt, wie man die Daten aus einer XML-Datei (siehe Listing 2) ausliest. Auch eine JDBC-konforme Datenbank kann als Datenquelle dienen. Der Rest des Programms erzeugt das Diagramm und zeigt es an (Abbildung 5). Mit der rechten Maustaste erscheint das abgebildete Kontextmenü. Abbildung 4 zeigt den Properties-Dialog, mit dem sich das Aussehen interaktiv manipulieren lässt.

Am Fließband

JFreechart erzeugt nicht nur interaktive Diagramme, sondern malt auch fertige Charts im Batch-Betrieb, die es als Jpeg oder PNG ausgibt. Wie das geht, zeigt wiederum Listing 1: Die Zeilen 122 bis 128 schreiben eine PNG-Datei sowie den HTML-Code für ein Imagemap.

Damit ist es möglich, auf bequeme Weise Reports im HTML-Format zu erstellen. Wer aktuelle Daten benötigt, bindet das Ganze in ein Servlet ein. Auch hierfür enthält JFreechart schon die notwendige Unterstützung sowie ein entsprechendes Beispiel. Darüber hinaus gibt es das Cewolf-Projekt [3], das sich

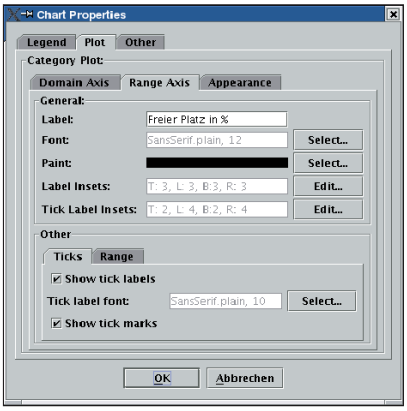

Abbildung 4: Der Eigenschaftsdialog erlaubt es, ein Diagramm vielfältig zu verändern und wie hier die Werteachse (Range Axis) im Detail einzustellen.

der Integration von JFreechart in JSP verschrieben hat. Da JFreechart auf das AWT aufsetzt, ist zumindest ein virtueller X-Server notwendig.

Weitere Ausgabeformate für JFreechart sind PDF und SVG. Ersteres nutzt die Itext-Bibliothek [4], Letzteres verwen-

Listing 1: »FreeSpace.java«

```
022: import java.io.*;
023:
024: import org.jfree.chart.*;
025: import org.jfree.chart.plot.*;
026: import org.jfree.data.*;
027: import org.jfree.data.xml.*;
028:
033: public class FreeSpace {
034:
035:   private static final String
036:     USB_STICK = "64MB USB-Stick",
037:     DVD_RAM = "4.3 GB DVD-RAM";
038:
039:   private CategoryDataset iSet;
040:
047:   private void createDataset() {
048:     DefaultCategoryDataset set = new
     DefaultCategoryDataset();
049:     set.addValue(91.81,"ext2",USB_STICK);
050:     set.addValue(98.94,"minix",USB_STICK);
051:     set.addValue(99.78,"vfat",USB_STICK);
052:     set.addValue(48.68,"reiserfs",USB_STICK);
053:     set.addValue(97.86,"jfs",USB_STICK);
054:     set.addValue(92.40,"xfs",USB_STICK);
055:
056:     set.addValue(93.43,"ext2",DVD_RAM);
057:     set.addValue(99.26,"reiserfs",DVD_RAM);
058:     set.addValue(99.57,"jfs",DVD_RAM);
059:     iSet = set;
060:   }
061:
068:   private void createDatasetFromXML(String
     filename) throws Exception {
069:     FileInputStream in = new
     FileInputStream(filename);
070:     iSet =
     DatasetReader.readCategoryDatasetFromXML(in)
     ;
071:     in.close();
072:   }
073:
080:   private JFreeChart createChart() {
081:     JFreeChart chart =
         ChartFactory.createBarChart(
082:       "Freier Platz auf leerem Linux FS",
083:       "Medium",  // Achsen-Label
084:       "Freier Platz in %", // Achsen-Label
085:       iSet,     // Datensatz
086:       PlotOrientation.VERTICAL,
087:       true,     // mit Legende
088:       true,     // mit Tooltips
089:       false);   // ohne URLs
090:     return chart;
091:   }
092:
100:   public static void main(String[] args) {
101:
102:     FreeSpace fs = new FreeSpace();
103:     String filename = null;
104:     boolean batchMode = false;
105:     try {
106:       if (args.length > 0) {
107:         if (args[0].equals("-b")) {
108:           batchMode = true;
109:           if (args.length > 1)
110:             filename = args[1];
111:         } else
112:           filename = args[0];
113:         }
114:         if (filename != null)
115:           fs.createDatasetFromXML(filename);
116:         else
117:           fs.createDataset();
118:
119:         JFreeChart chart = fs.createChart();
120:
121:         if (batchMode) {
122:           ChartRenderingInfo info = new
     ChartRenderingInfo();
123:           ChartUtilities.
124:             saveChartAsPNG(new
     File("freespace.png"),chart,600,400,info);
125:           PrintWriter pw = new
     PrintWriter(System.out);
126:           ChartUtilities.
127:             writeImageMap(pw,"FreeSpace",info);
128:           pw.flush();
129:         } else {
130:           ChartFrame frame = new
     ChartFrame("FreeSpace",chart);
131:           frame.pack();
132:           frame.setVisible(true);
133:         }
134:       } catch (Exception e) {
135:         e.printStackTrace();
136:       }
137:     }
138: }
```

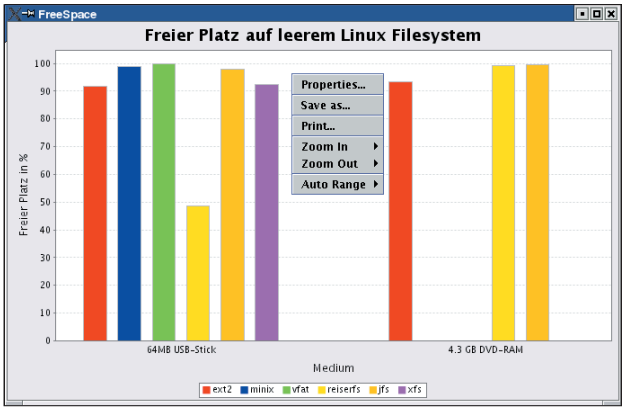

Abbildung 5: Eine mit JFreechart geschriebene Beispielanwendung (Listing 1) zeigt den freien Speicherplatz eines USB-Sticks und eines DVD-RAM nach dem Formatieren: ReiserFS und JFS schneiden besser ab als Ext 2.

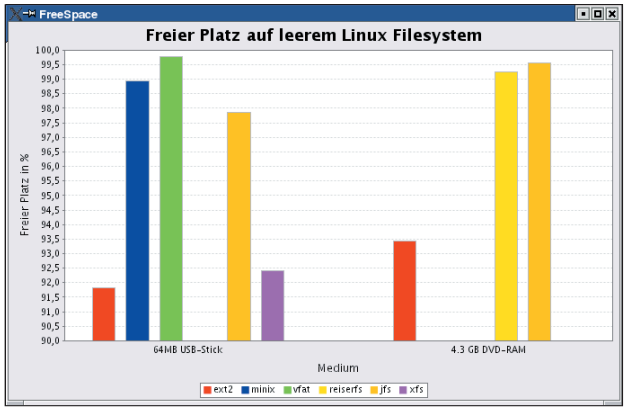

Abbildung 6: Ein Diagramm mit denselben Werten, die Abbildung 4 zugrunde liegen. Die nicht-proportionale Darstellung lässt den Vorteil von Minix und FAT größer erscheinen, als er tatsächlich ist.

det das Batik-Toolkit [5]. Beide stellen jeweils eine »Graphics2D«-Implementation zur Verfügung, mit der JFreechart die Diagramme zeichnet. Ein Artikel in Dev-X [6] beschreibt, wie man aus beliebigen XML-Dateien über XPath die Eingabedateien für JFreechart erzeugt.

finally{}

JFreechart ist eine sauber aufgebaute Bibliothek – von simplen Interfaces bis zu mächtigen GUI-Komponenten – und deshalb einfach zu verwenden. Trotzdem sollte man berücksichtigen, dass ein Diagramm zwar schnell erzeugt ist, aber oft doch nur sinnlos Platz verschwendet. Beispiele dafür gibt es genug. Ein Diagramm, das man erklären oder interpretieren muss, hat seinen Zweck verfehlt. Abschreckend soll Abbildung 6 wirken. In dieser Abbildung entsteht der Eindruck, dass Minix und FAT ein Vielfaches des freien Platzes bieten, den das Ext-2-System bereitstellt.

Der Grund dafür ist, dass die Länge der Balken nicht proportional zu den repräsentierten Werten ist. Unser Auge reagiert aber auf den Schein und interpretiert die Proportionen der Balken. Wie man sieht, ist es gar nicht nötig, nach der alten Statistiker-Weisheit „Traue keiner Statistik, die du nicht selbst gefälscht hast!“ zu handeln. Ein geeignetes Diagramm tut’s auch. *(ofr)* ■

Der Autor

Bernhard Bablok arbeitet bei der AGIS mbH als Anwendungsentwickler. Wenn er nicht Musik hört, mit dem Radl oder zu Fuß unterwegs ist, beschäftigt er sich mit Themen rund um Objektorientierung. Er ist unter [coffee-shop@bablokb.de] zu erreichen.

Infos

[1] Freechart-Homepage: [http://jfree.org/jfreechart]

[2] Homepage von Object-Refinery: [http://www.object-refinery.com]

[3] Homepage des Cewolf-Projekts: [http://cewolf.sourceforge.net]

[4] I-Text, eine Bibliothek für die Erstellung von PDFs: [http://www.lowagie.com/iText]

[5] Batik, das Apache-SVG-Projekt: [http://xml.apache.org/batik]

[6] Laurence Moroney: Serve Business Graphics from Any XML Source: [http://www.devx.com/xml/Article/7956]

Listing 2: Eingabedaten für ein Balkendiagramm

```
01: <?xml version="1.0" encoding="UTF-8"?>
02:
03: <!-- Data for FreeSpace -->
04:
05: <CategoryDataset>
06:   <Series name = "ext2">
07:     <Item>
08:       <Key>64MB USB-Stick</Key>
09:       <Value>91.81</Value>
10:     </Item>
11:     <Item>
12:       <Key>4.3 GB DVD-RAM</Key>
13:       <Value>93.43</Value>
14:     </Item>
15:   </Series>
16:
17:   <Series name = "minix">
18:     <Item>
19:       <Key>64MB USB-Stick</Key>
20:       <Value>98.94</Value>
21:     </Item>
22:   </Series>
23:
24:   <Series name = "vfat">
25:     <Item>
26:       <Key>64MB USB-Stick</Key>
27:       <Value>99.78</Value>
28:     </Item>
29:   </Series>
30:
31:   <Series name = "reiserfs">
32:     <Item>
33:       <Key>64MB USB-Stick</Key>
34:       <Value>48.68</Value>
35:     </Item>
36:     <Item>
37:       <Key>4.3 GB DVD-RAM</Key>
38:       <Value>99.26</Value>
39:     </Item>
40:   </Series>
41:
42:   <Series name = "jfs">
43:     <Item>
44:       <Key>64MB USB-Stick</Key>
45:       <Value>97.86</Value>
46:     </Item>
47:     <Item>
48:       <Key>4.3 GB DVD-RAM</Key>
49:       <Value>99.57</Value>
50:     </Item>
51:   </Series>
52:
53:   <Series name = "xfs">
54:     <Item>
55:       <Key>64MB USB-Stick</Key>
56:       <Value>92.40</Value>
57:     </Item>
58:   </Series>
59: </CategoryDataset>
```