Java-Programmierung mit Visual J++ 1.1

Dieter Reiermann

Das vorliegende Fachbuch soll sowohl einen technischen Einstieg für Entwickler als auch einen Überblick zur Verbesserung des Up To Date Wissenstandes von einschlägig interessierten Managern geben . Vorwissen zu OOP (objektorientierter Programmierung) ist nicht unbedingt erforderlich (Vorwort, sinngemäß).

Zunächst ein Überblick über Gliederung und Inhalt

Das Buch ist in eine Einleitung, 6 Abschnitte und ein Stichwortverzeichnis gegliedert.

Im ***ersten Kapitel*** („Das Foyer") werden grundlegende Begriffsbestimmungen (z.Bsp. Objekte, Vererbung, OO-Design) erklärt, außerdem enthält es die Zusammenfassungen der folgenden 5 Kapitel.

Alle Kapitel werden - einem aktuellen Trend folgend - von „FAQs" (Frequently Asked Questions And Answers) eingeleitet.

Der letzte Abschnitt des ersten Kapitels zeigt die Intentionen des Autors - ist also eine Art Generalzusammenfassung:

„Der Autor hofft, Ihnen mit diesem Buch einen praktischen Ratgeber an die Hand zu geben, der Ihnen einige Aha-Effekte vermittelt und Ihnen hilft, sich in der teilweise verwirrenden Diskussion um Java und die neuen Web-Techniken besser zurechtzufinden. Durch die Herstellung des Zusammenhanges zwischen Visual J++,, Java und OO-Ansätzen in praktischen, nachvollziehbaren Beispielen werden Erkenntnisse, die Sie aus diesem Buch gewinnen, den Produktlebenszyklus von Visual J++ hoffentlich überleben."

Kapitel 2 Die Bausteine des Programmiersystems, die Programmiersprache Java, Klassenbibliotheken, der Java-Compiler, die Java Virtual Machine, die Bestandteile von Visual J++ und Windows und Java.

Im ***Kapitel 3*** wird ein erstes, sehr einfaches Java-Applet (Versandkostenberechnung) mit Visual J++ entwickelt. Um einen ersten praktischen Eindruck von Visual J++ zu bekommen, ist es ratsam, die Entwicklung des Applets nachzuvollziehen. Deswegen möchte ich, der ich damit meine ersten Gehversuche in Java begonnen habe, diesen Abschnitt etwas genauer beleuchten.

Zunächst steht man etwas ratlos vor dem Bildschirmarbeitsplatz eines offensichtlich mächtigen Entwicklungssystems.

Doch schön langsam, Schritt für Schritt erklärt der Autor, wie ein Projekt angelegt, der Applet Wizard eingestellt wird, das Klassensystem im Class View-Fenster und das Dateisystem im FileView-Fenster benützt wird, um Methoden und Variablen zu den automatisch erzeugten Source-Rahmen hinzuzufügen. Die Benutzeroberfläche (der Dialog) wird angelegt - hier wird man an Visual Basic erinnert. Mit dem Resource-Editor werden die Steuerelemente plaziert und ihre Eigenschaften bestimmt, der Resource-Wizard generiert daraus eine Java-Quellcode-Datei. Die Benutzeroberfläche (die Klasse `VersandkostenDialog`") wird nun mit der (noch leeren) funktionellen Klasse `Versandkosten` verbunden. Das Auswahllisten-Control aus dem Dialog muß nachträglich mit den Auswahlelementen gefüttert werden. Daran ist, wie uns der Autor erklärt, ein Fehler des Resource-Wizard schuld.

Hier wird erstmalig Quellcode - sozusagen händisch - eingefügt. Die mit Unterstützung des Resource-Wizards entstandenen Quellcode-Dateien (`VersandkostenDialog.java` und `DialogLayout.java`) werden nun mit `Insert Files Into Project` dem Projekt hinzugefügt. Die beiden Klassen `Versandkosten` und `VersandkostenDialog` stehen nun zwar im Projekt, müssen aber noch mit `import VersandkostenDialog` mit der Klasse `Versandkosten` verbunden werden.

Nun muß noch die auf den Mausklick reagierende Event-Methode und der Zugriff auf den Inhalt der Steuerelemente in die Versandkostenklasse eingebaut werden. Dazu wird eine Methode `action()` geschaffen, die auch gleich die Berechnung der Versandkosten enthält:

```
public boolean action(Event evt, Object what) {
  if (what.equals(„OK")) {
    double Grundpauschale = 0.0;
    Double VersandkostenErgebnis = new Double(0.0);
    if
(Integer.parseInt(dlg.IDC_Wahrenwert.getText())t)
    //Grundpauschale nach Land ermitteln//
    switch (dlg.IDC_Zielland.getSelected(Index()) {
    case 0:
      Grundpauschale=10.0;
      break;
    case 1:
      Grundpauschale=20.0;
      break;
    case 2:
      Grundpauschale=30.0;
      break;
    }
    //Aufschlag für Express-Versand berechnen//
    if (dlg.IDC_CHECK1.getState()) Grundpauschale
*=1.25;
  }
  VersandkostenErgebnis=new Double(Grundpauschale);
  dlg.IDC_Versandkosten.setText
(VersandkostenErgebnis.toString());
  // Event handled //
  return true;
  }
  return false;
}
```

Jetzt kann das Applet entweder im Browser (zB. Microsoft Internet Explorer) oder direkt in der Oberfläche getestet werden.

Kapitel 4 beschreibt fortgeschrittene Konzepte und zeigt die Erweiterbarkeit des Versandkosten-Applets.

Kapitel 5 widmet sich den möglichen Umgebungen, in denen ein Visual J++ Applet laufen kann (Internet, Desktop, Java-Server etc.).

Kapitel 6 führt zum absoluten Höhepunkt einer Applet-Entwicklung: Teile des Applets laufen auf dem Internet-Server, der Rest im Webbrowser des Clients.

Mein (erster) Eindruck: Ohne praktische Grundkenntnisse in OOP und C++ wird man es bei der Lektüre dieses Buches nicht leicht haben.

Trotzdem: Wer sich komplexe neue Programmentwicklungswerkzeuge mit „learning by doing" gut erarbeiten kann, dem ist dieses Buch zu empfehlen.

Java verstehen und effektiv nutzen, Torsten Schlabach, Addison-Wesley-Longman, 1997, 359 Seiten, 1 Diskette, ISBN 3-8237-1191-8, ATS 364