FileMaker Erweiterung

Einführung in die Anwendung der Plugin-Technik

Plugins produktiv einsetzen (Teil 1)

Nicolaus Busch, Uster (Schweiz)

Als FileMaker mit Version 4 die Plugin-Technik einführte, ging dies zunächst etwas unter. Zu sehen gab es anfangs nur ein einziges Plug-in, den Webcompanion, und wer ahnte damals schon, wie grundlegend die Technik der Webanbindung die Positionierung von FileMaker verändern sollte?

Inzwischen hat sich einiges getan. Eine Suche in der Plug-in-Liste bei FileMaker, Inc. (***www.filemaker.com/products/search_plugins.html***) fördert nicht weniger als 178 verschiedene Plugins zutage, die sich so unterschiedlicher Aufgaben wie dem Auslesen von Daten über die serielle Schnittstelle oder der Erstellung von Tortengrafiken widmen. Wir wollen in den nächsten Ausgaben des FileMaker-Magazins eine subjektive Auswahl von Plugins vorstellen und dabei aufzeigen, wie Sie die Plugins sinnvoll einsetzen können. Den Anfang machen wir hier mit einer allgemeinen Einführung in das Arbeiten mit Plugins sowie einer Vorstellung des **Troi Coding-Plugins**, das die Kodierung besonders sensibler Daten erlaubt.

Was ist ein Plugin?

Ein Plugin ist zunächst einmal ein Programm, also kompilierter Code, der von einem Rechner ausgeführt werden kann. Es handelt sich um eine Datei (auf Windows-Rechnern mit der Endung ***.FMX***), die FileMaker in einem speziellen Ordner (Windows: ***System***; Macintosh: ***FileMaker Erweiterungen***) erwartet. Sie finden diese Ordner immer dort, wo Sie auch das Programm FileMaker Pro selbst finden. Obwohl es erfreulich viele Plugins sowohl für Macintosh als auch für Windows gibt, ist es wichtig zu wissen, dass es sich stets um verschiedene Versionen handelt. Sie können nie ein Plugin einfach von Windows nach Mac kopieren oder andersherum, sondern benötigen immer die für das jeweilige System kompilierte Version.

Beim Programmstart scannt FileMaker den Ordner ***System*** (Windows) bzw. ***FileMaker Erweiterungen*** (Macintosh) nach Plugins und lässt sich von jedem die Funktionsliste übermitteln. Alle diese Funktionen finden Sie im Formeleditor unter „Externe Funktionen“ (Bild 1).

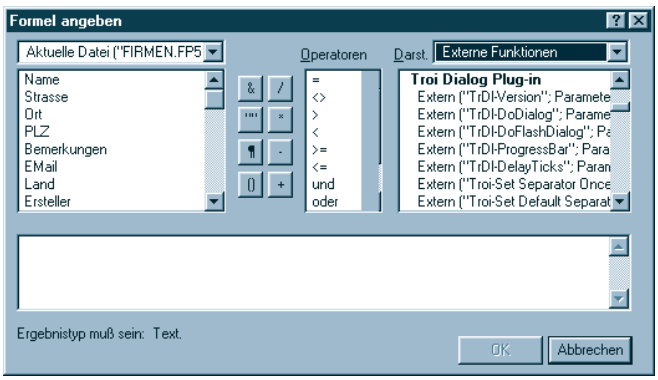


Abb. 1 Liste der externen, aus dem Plugin stammenden Funktionen

Beachten Sie, dass nur hier alle verfügbaren externen Funktionen in ihrer korrekten Schreibweise aufgelistet sind. Wählen Sie statt dessen die normale Darstellung („Alle nach Namen“), so zeigt der Formeleditor nur die allgemeine Form

```
Extern (Name; Parameter)
```

Dies ist wenig hilfreich, zumal es beim Aufruf von Plugins auf die ganz korrekte Groß-/Kleinschreibung ankommt – bei dem Befehl wie

```
Extern
    ("TrDl-DoDialog"; Parameter)
```

nimmt man da ganz gerne einmal die Hilfe des Systems in Anspruch.

Ein Plugin kann also mehrere Funktionen zur Verfügung stellen. Was diese Funktionen anstellen können, findet seine Grenzen im Prinzip nur in der Phantasie der Entwickler – was sich auf dem jeweiligen System programmieren lässt, kann theoretisch auch ein Plugin erledigen. Der Unterschied zu einem normalen Programm besteht lediglich darin, dass Plugins von FileMaker Parameter entgegennehmen und diese zurückgeben können.

Wie funktioniert ein Plugin?

Wie das Plugin intern arbeitet, kann uns als FileMaker-Entwicklern vollständig egal sein (wenn wir einmal voraussetzen, dass es sauber programmiert ist und nicht etwa die Festplatte unseres besten Kunden formatiert). Wichtig ist, wie die Zusammenarbeit mit FileMaker aussieht. Zum Glück ist die Regelung hier ganz einfach: Ein Plugin empfängt genau einen Parameter und gibt genau einen Rückgabewert zurück. Durch den Aufruf der Funktion wird gleich festgelegt, wo dieser Rückgabewert abgelegt werden soll. Schauen wir uns das einmal am Beispiel an, das übrigens nicht aus einem Programm von Microsoft stammt.

```
◆ Feld angeben
Feld: MeinDatenFeld
Formel:
      Extern ("TrDl-DoDialog";
      "Änderungen nicht speichern?
      |Weiter|OK|Ja|Fortfahren")
```

Der Aufruf einer externen Funktion, eines Plugins also, besteht immer aus diesen drei Elementen: einem Feld, welches das Ergebnis entgegennimmt (**MeinDatenFeld**), einer Funktion ("TrDl-DoDialog") und einem Parameter, der an das Plugin übergeben wird. Wenn Sie sich obiges Beispiel ansehen, werden Sie sofort bemerken, dass dieser Parameter offenbar mehrere Argumente enthält, die durch das Zeichen „|“ getrennt werden.

Dies hat sich zwar als eine Art Standard etabliert, liegt aber tatsächlich völlig im Ermessen des Plugin-Entwicklers. Was er mit dem von FileMaker übergebenen String anstellt, ist alleine seine Angelegenheit. Er muss lediglich irgendein Ergebnis zurückgeben, damit FileMaker seine Arbeit fortsetzen kann.

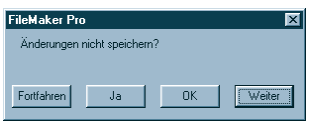


Abb. 2 So sieht das Ergebnis der Funktion aus

Im Beispiel liefert Ihnen das **Troi Dialog-Plugin** die Nummer und den Wert der gedrückten Taste zurück. Hat der Anwender auf „Ja“ geklickt, so haben Sie bei →

obigem Aufruf anschließend den Eintrag „3“ in Ihrem Feld **MeinDatenFeld** stehen. Das **Troi Dialog-Plugin** kann noch sehr viel mehr, wie Sie in dem jetzt folgenden Artikeln in dieser Ausgabe werden feststellen können.

Was kann ein Plugin?

Zusammenfassend kann ein Plugin also eine oder mehrere beliebige Funktionen ausführen, dabei Daten von FileMaker entgegennehmen und verarbeiten, und das Ergebnis dieser Funktionen in ein FileMaker-Feld zurückschreiben.

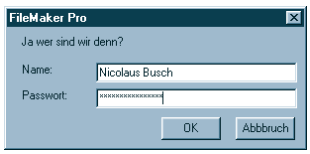
Ein Plugin kann dabei auch FileMaker-Funktionen benutzen, soweit diese für externe Aufrufe zugänglich sind, es kann System-Funktionen nutzen und es kann beliebige andere Programme aufrufen. Daraus wird aber auch deutlich, was nicht geht: Manipulationen der Datenstruktur etwa, die FileMaker verbietet, sind nicht möglich, und auch ein senkrechtes Portal wird ein Plugin nicht realisieren können.

Aber dafür kann es nette Sachen wie diese:

Nach diesem Aufruf in einem Scriptschritt

◆ Feld angeben
Feld: *MeinDatenFeld*
Formel: *Extern ("TrDl-DoDialog"; "Ja wer sind wir denn? | Abbruch | OK | | | Userpassword | " & Status (AktuellBenutzername) & " | Name: | Passwort: ")*

sehen Sie dieses Eingabefenster:



Die eigene Passwortverwaltung in der FileMaker-Lösung mit automatischer Sternchenschrift ist Dank **Troi Dialog-Plugin** nun kein Problem mehr! ■

FileMaker Erweiterung

Das Troi Coding Plugin

Plugins produktiv einsetzen (Teil 2)

Nicolaus Busch, Uster (Schweiz)

Wenn Sie sich einmal die Mühe machen, Ihre mit einem raffinierten Passwortsystem ausgestattete Datenbank in einem Hexeditor zu betrachten, dann sehen Sie so etwas:

```
3C 0005 0358 C101 ........1...X..
74 6D61 6E6E 0263 .T.th.hartmann.c
38 4004 0000 01F5 hemie.gmbh@.....
35 6F70 7469 6D69 ..therapieoptimi
3E 5EC0 CA74 6875 erung@....^..thu
7F 0400 0001 F6C0 eringer.........
35 696D 6974 7465 .tierarzneimitte
74 6D74 02FF FF04 l@.....tmt....
39 6B6F 6C6F 6769 ......toxikologi
74 7265 6E6E 7465 e@....u..trennte
```

Von wegen kein Lesezugriff …

Nicht gerade schön und auch nicht einfach zu lesen, aber: Jede in FileMaker gespeicherte Information steht in Klarschrift auf der Festplatte und ist nicht gegen fremde Zugriffe geschützt. Hier tritt das **Troi Coding Plugin** auf den Plan und macht Ihre Texte unlesbar.

Den Text `Meine ganz geheime Information` legt es z.B. so auf der Platte ab: `§ â q¡Ö ‹¥ÔÍ_[…”%x']‹´Øª:Iî4!¢Í ±-_ ˆ±Ê/_áfR‹JK Q`, und Sie müssen zugeben, dass man nun so seine Schwierigkeiten mit dem Lesen hat. Bevor ich anhand eines Projektbeispieles die einzelnen Schritte zur Verwendung des Plugins durchgehe, schauen wir uns einmal den zentralen Befehl an.

So sieht der Aufruf des Plugins aus, der den Inhalt des Feldes **Geheiminfo** verschlüsselt:

◆ Feld angeben
Feld: *Geheiminfo*
Formel: *Extern ("Troi-Code"; "-encryptDES | " & V_Schluessel & " | " & Geheiminfo)*

Wie in der Einführung (siehe Seite 7) erläutert, kann ein Plugin immer nur einen Parameter entgegennehmen. Dieser wird hier aus drei Teilen zusammengefügt, die wieder durch die senkrechten Striche separiert werden. Dabei ist es dem Plugin egal, ob Sie den Parameter als Text direkt eingeben oder erst, wie hier gezeigt, von FileMaker aus Feldern zusammenbauen lassen. Als Parameter übergeben Sie hier einen Schalter ("-encryptDES"), einen Schlüssel und den zu verschlüsselnden Text.

Wollen Sie das ganze rückgängig machen, ändern Sie nur den Schalter:

◆ Feld angeben
Feld: *Geheiminfo*
Formel: *Extern ("Troi-Code"; "-decryptDES | " & V_Schluessel & " | " & Geheiminfo)*

Im Folgenden möchte ich alle Aufrufe des Plugins durchgehen, die ich in einem Projekt benutzt habe. Damit bleibt etwa die Hälfte der Funktionen des Plugins unerwähnt, aber es gibt schließlich eine Dokumentation.

Eines sollte man aber vorwegschicken, bevor Sie lange herumprobieren: die Verschlüsselungsfunktion arbeitet nur mit Textfeldern, nicht mit Feldern vom Typ Zahl! Zahlen in Textfeldern werden aber völlig korrekt behandelt.

Vorarbeiten

Anders als bei anderen Plugin-Entwicklern gibt es bei der Firma Troi keine zwei verschiedenen Dateien für die Demo- und die Vollversion: Die Demoversion wird stattdessen durch einen Schlüssel freigeschaltet, den Sie beim Kauf einer Einzelplatz- oder einer Entwicklerlizenz erhalten. Zwei verschiedene Wege stehen zur Verfügung:

Sie können die Registrierschlüssel für alle Plugins in eine Textdatei schreiben und diese unter dem Namen ***Troipay.txt*** in Ihr FileMaker-Verzeichnis legen, oder Sie schicken den Registrierschlüssel direkt aus der Datenbank an das Plugin, bevor Sie eine seiner Funktionen benutzen. Ich selbst favorisiere den zweiten Weg, weil die Kunden sehr schnell einmal eine Datei löschen.

Das Plugin wird registriert, indem die Funktion „TrCo-Version“ aufgerufen wird. Als Parameter übergeben Sie den Registrierungsschlüssel, den man am Besten in eine Variable schreibt. Da die Lizenzen und damit die Schlüssel für Mac und Windows unterschiedlich sind, benutzen wir die zwei Variablen **V_WinRegTroi** (Typ Text) und **V_MacRegTroi** (Typ Text).

Die Variable **V_PluginAntwort** nimmt entgegen, was immer die diversen Plugins an Rückgabecode auswerfen.

So registriert man das Plugin:

◆ Feld angeben
Feld: *V_PluginAntwort*
Formel: *Extern ("TrCo-Version"; Falls (Status(AktuellPlattform)= 1; V_MacRegTroi; V_WinRegTroi))*

Wenn Sie möchten, können Sie an dieser Stelle den Rückgabewert in **V_PluginAntwort** auswerten, das ist aber nicht zwingend; die Registrierung erfolgt stillschweigend, Sie bemerken den Misserfolg beim ersten Aufruf einer Funktion des Plugins, das Sie dann auf die fehlende Registrierung hinweist. Das Feld beinhaltet eine „0“, wenn die Registrierung erfolgreich war, andernfalls eine Zahl größer als „1“.

Und wenn das Plugin nun gar nicht installiert ist? Verständlich, dass das Plugin Ihnen dann keine Fehlermeldung ausgeben kann, unverständlich allerdings, dass FileMaker dies nicht kann. Dem ist es nämlich vollständig

egal, was Sie da so treiben, und so wird selbst dieser Funktionsaufruf nicht nur vom ScriptMaker akzeptiert, sondern auch ohne jede Reaktion ausgeführt:

```
◆ Feld angeben
   Feld:    V_PluginAntwort
   Formel:  Extern ("Rudolf das
            rotnasige Rentier";
            "trinkt gerne Punsch")
```

Es ist also eine gute Idee, das Vorhandensein benötigter Plugins sicherzustellen. Sie erreichen dies bei allen Plugins von Troi durch die Funktion „TrXX-Version“, wobei „XX“ für das jeweilige Plugin steht:

```
◆ Wenn                                    ▼
   Formel: Links (
           Extern ("TrCo-Version"; "");
           19) <> "Troi Coding Plug-in")
◆ Meldung
   Text:    Sie haben das Coding-
            Plugin nicht installiert.
            Bitte informieren Sie Ihren
            Supervisor.
   Taste:   OK
◆ Programm beenden                        ◆
◆ Ende-Wenn                               ▲
```

Wie Sie sehen, brauchen Sie nicht immer den Scriptschritt „Feld angeben“ zu benutzen, Sie können die Rückgabewerte eines Plugins direkt in einer Berechnung verwenden.

Nun gut, das Plugin ist vorhanden und registriert, nun wollen wir endlich etwas codieren. Konkret sollen einige personenbezogene Daten wie die Höhe des Einkommens nicht für jeden sichtbar sein, der die Datenbank benutzt. Fragt sich nur, mit welchem Kennwort codieren wir? Es tauchen drei zusammenhängende Probleme auf:

- Der Anwender will und soll nicht jedes Mal ein Kennwort eingeben müssen;
- Wird zum Verschlüsseln das falsche Kennwort benutzt, können die Daten nie wieder entschlüsselt werden;
- Das Kennwort darf nicht unverschlüsselt in der Datenbank gespeichert werden, sonst wäre die ganze Aktion sinnlos.

Das Kennwort soll also nur einmal pro Sitzung eingegeben werden. Damit sichergestellt ist, das es selbst bei einem Stromausfall nicht in der Datenbank zu finden ist, schreiben wir es in eine Variable – im Netzbetrieb werden Variablen pro User geführt und nur im Arbeitsspeicher gehalten, aber nicht in die Datenbank geschrieben. Um sicherzustellen, dass nicht ein Tippfehler die Daten mit dem falschen Kennwort codiert, benutzen wir wiederum eine Funktion des Plugins.

Unverzichtbare Zugaben

Es gehört zu den Qualitätsmerkmalen der Troi-Plugins, dass die hilfreichen Zusatzfunktionen immer schon integriert sind. Mit Hilfe der Checksummen-Funktion haben wir einmalig unser Passwort geprüft und das Ergebnis in eine Globaldatei geschrieben:

```
◆ Feld angeben
   Feld:    ChiffreChecksum
   Formel:  Extern ("Troi-Checksum";
            "blumentopferde")
```

Gegen diesen Wert prüfen wir jetzt das vom Anwender eingegebene Passwort:

```
◆ Schleife                                ▼
◆ Feld angeben
   Feld:    V_Chiffre
   Formel:  Extern ("TrDl-DoDialog";
            "¶Bitte geben Sie das
            Chiffrierkennwort ein|OK
            |Abbruch|||Password
            ||Kennwort:")"
◆ Schleife-Verlass.-Wenn                  ◆
   Formel:  Extern ("Troi-Checksum";
            V_Chiffre) = ChiffreChecksum
◆ Meldung
   Text:    Falsches Kennwort!
            Nochmal versuchen?
   Taste 1: Ja
   Taste 2: Nein
◆ Wenn                                    ▽
   Formel:
            StatusAktuellMeldungsWahl = 2
◆ Programm beenden                        ◇
◆ Ende-Wenn                               △
◆ Ende-Schleife                           ▲
```

Die hier benutzte Funktion „Troi-Checksum“ ermittelt für einen Text eine ganzzahlige Prüfsumme zwischen 1 und 1024. Für unser Passwort „blumentopferde“ generiert es z.B. die Zahl 476.

Wenn ich auch oben angekündigt hatte, die hier nicht benutzten Funktionen außen vor zu lassen, sollte man doch erwähnen, dass das **Coding-Plugin** noch solche Nettigkeiten wie „Text-Signature“ (prüfen, ob eine Textfeld verändert wurde) oder „NumToBinary“ (Zahl in ihren Binärwert umrechnen) mitbringt.

Das Finale?

Jetzt aber wirklich! Das Plugin ist vorhanden und registriert, das Kennwort eingegeben und geprüft. Jetzt können wir per Script die gewünschten Felder chiffrieren:

```
◆ Feld angeben
   Feld:    Jahreseinkommen
   Formel:  Extern ("Troi-Code";
            "-encryptDES|" &
            V_Chiffre & "|" &
            Jahreseinkommen)
◆ Feld angeben
   Feld:    Zulagen
   Formel:  Extern ("Troi-Code";
            "-encryptDES|" &
            V_Chiffre & "|" & Zulagen)
```

Dieses Script können Sie auf einen Knopf legen, und nun nach Herzenslust geheime Informationen produzieren. Bevor nun alle wild durcheinander rufen und uns für ihren Datenverlust verantwortlich machen, schreiben wir dies lieber in fett:

Machen Sie zuerst ein Backup!

Bitte? Ja, sind Sie denn nie zufrieden? Also gut, wir haben zwei Probleme produziert:

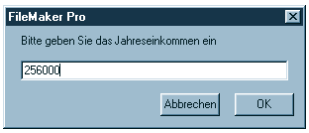
- Während der Eingabe ist der Feldinhalt uncodiert und kann von jedem gelesen werden, der an die Datei herankann.
- Das statistische Mittel der Werte „**)!§&%=**“, „**§%=§(**“ und „***'_:;/(**“ ist vergleichsweise schwer zu ermitteln (unter uns, es ist 42).

Man braucht also Verfahren, um einerseits den Text von vornherein codiert ablegen zu können und andererseits, um alle Datensätze auf einen Schlag decodieren zu können.

Das Grande Finale

Hatte ich schon erwähnt, dass man Plugins nicht immer via „Feld angeben“ aufrufen muss? Man muss sie auch nicht einzeln aufrufen, und deshalb löst dieses Script das erste Problem:

```
◆ Feld angeben
   Feld:    Jahreseinkommen
   Formel:  Extern ("Troi-Code";
            "-encryptDES|" &
            V_Chiffre & Extern
            ("TrDl-DoDialog";
            "Bitte geben Sie das
            Jahreseinkommen
            ein|OK|Abbrechen
            |||input"))
```



Das Gehalt wird bei dieser Methode überhaupt nie in Klarschrift in die Datenbank geschrieben, sondern vom **Dialog-Plugin** direkt an das **Coding-Plugin** weitergereicht. Wenn Sie auf die Auswertung verzichten können und lediglich die Möglichkeit brauchen, die codierte Information nachschlagen zu können, dann können Sie auch die Decodierungsmöglichkeiten auf die Anzeige per Plugin beschränken; der Wert taucht dann niemals in Klarschrift auf. *(Eine Herausforderung für Tüftler: Übergeben Sie den Wert zunächst an das Plugin* **oAzium Charts**, *lassen Sie eine Tortengrafik erstellen und codieren Sie dann die Grafik!).*

Das zweite Problem kollidiert natürlich erheblich mit dieser Lösung. Wollen Sie eine Statistik über Daten erzeugen, dann müssen diese unchiffriert in der Datenbank vorliegen. Zum Glück brauchen Sie aber auch dies nicht per Hand zu erledigen. Rufen Sie die entsprechenden Datensätze auf und decodieren Sie alles mit einem Script dieser Art:

```
◆ Ersetzen
   Feld:    Jahreseinkommen
   Formel:  Extern ("Troi-Code";
            "-decryptDES|" & V_Chiffre
            & "|" & Jahreseinkommen)
◆ Ersetzen
   Feld:    Zulagen
   Formel:  Extern ("Troi-Code";
            "-decryptDES|" &
            V_Chiffre & "|" & Zulagen)
```

Sie sehen also, dass man mit dem **Troi Coding Plugin** ohne allzu grossen Aufwand einzelne Felder verschlüsseln kann. Deutlich sollte aber auch geworden sein, dass FileMaker dadurch nicht zu einer Hochsicherheits-Datenbank wird. Zur Verwaltung von Daten, die unabhängig von den Kosten keinesfalls in die Hände Dritter gelangen dürfen, ist FileMaker nicht das geeignete Tool. ■

FileMaker Magazin

Abo-Bestellung

☐ Ja! Ich will das **FileMaker Magazin** abonnieren. Für 95 Mark (europ. Ausland: 105 Mark) bekomme ich sechs Ausgaben pro Jahr (inkl. MwSt., Porto und Versand). Das Abonnement gilt für mindestens ein Jahr und verlängert sich um ein weiteres Jahr, wenn ich nicht einen Monat vor Ablauf schriftlich kündige.

Das Abonnement soll mit der Ausgabe _____ beginnen. *(Rückwirkender Abobeginn ist möglich!)*

Garantie
Diese Bestellung kann innerhalb von 10 Tagen schriftlich beim K&K Verlag widerrufen werden. Zur Fristeinhaltung genügt die Absendung des Widerrufs innerhalb dieser 10 Tage (Poststempel).

✗ .....
Datum und Unterschrift zur Kenntnisnahme des Widerspruchsrechts

---

FMM Beispieldateien–Abonnement

☐ Senden Sie mir ab sofort bis auf Widerruf zu jeder Ausgabe auch die **FMM Beispiel-Dateien**:

☐ als **E-mail** für ☐ Windows / ☐ Macintosh
☐ auf **Diskette** (nur Macintosh)

Den Betrag von 60 Mark (europ. Ausland: 75 Mark) für die ersten sechs Ausgaben bezahle ich wie unten angegeben.

Bestellung Jahrgänge

☐ Ich bestelle den **Jahrgang 1999** für 75 Mark (europ. Ausland: 85 Mark)
☐ Ich bestelle den **Jahrgang 1998** für 55 Mark (europ. Ausland: 65 Mark)
☐ Ich bestelle den **Jahrgang 1997** für 35 Mark (europ. Ausland: 45 Mark)
☐ Ich bestelle den **Jahrgang 1996** für 15 Mark (europ. Ausland: 25 Mark)
☐ Ich bestelle den **Jahrgang 1995** für 5 Mark (europ. Ausland: 15 Mark)
Ich erhalte die Jahrgänge nach Zahlungseingang portofrei zugesandt.

Gewünschte Zahlungsart

☐ **Bankeinzug** *(leider nur im Inland möglich):*

....
Geldinstitut *Bankleitzahl* *Kontonummer*

☐ **Kreditkarte** ○ Visa / ○ EuroCard
Karten-Nr. ☐☐☐☐
Gültig bis ☐☐/☐☐

☐ **Verrechnungsscheck**. Der Scheck liegt dieser Bestellung (als Brief) bei.
☐ **Rechnung**. Bitte schicken Sie mir vorab eine Rechnung. Die Ware erhalte ich, sobald meine Zahlung auf dem Konto des K&K Verlag eingegangen ist.

.....
Firma / Name (Bitte in Druckbuchstaben!)

.....
Straße

....
Land *Postleitzahl* *Ort*

....
Telefon *Telefax* *E-mail*

✗ .....
Datum und Unterschrift

K&K Verlag GmbH, Mörkenstraße 12, 22767 Hamburg · Fax: +49 (0)40/38 61 08 70 · abo@filemaker-magazin.de