Pro Advanced mittel

(Human) Interface Design (Teil 2)

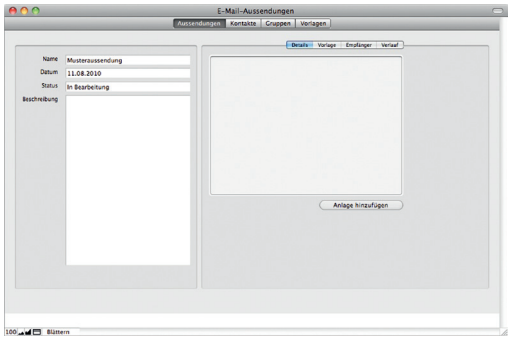
Der Dialog zwischen Mensch und Anwendung

Holger Darjus

(Jg. 1962) ist Technischer Leiter des Medienvorstufen-Unternehmens Einsatz Creative Production GmbH & Co. KG in Hamburg mit rund 45 Mitarbeitern. Er entwickelt seit 1996 FileMaker Datenbanken speziell für die Medienvorstufe.

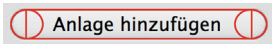
h.darjus@einsatz.de

Im ersten Teil dieser kleinen Serie habe ich eine grundsätzliche Überlegungen zum Thema Interface Design angestellt und damit begonnen, die FileMaker Beispieldatei *E-Mail-Aussendungen.fp7* entsprechend zu gestalten. Der letzte Stand sah wie folgt aus:



In diesem Teil werde ich mich zunächst dem hässlichen „Schatten" widmen, den runde Tasten beim Anklicken bekommen.

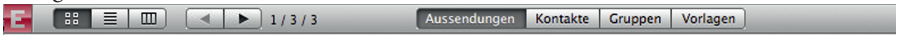
Mithilfe eines kleinen Tricks kann man sich einen solch unschönen Anblick in Zukunft ersparen: Statt die Grafik selbst als Taste zu benutzen, habe ich mit den grafischen Werkzeugen von FileMaker aus zwei Kreisen und einem Rechteck eine passende Form erstellt, die ich über die Grafik lege:



Nun muss nur noch die Linienstärke auf „0" gestellt werden und schon sind die Voraussetzungen für optisch einwandfrei funktionierende Tasten geschaffen. Diese sind in der Erstellung zwar etwas kompliziert, aber dafür werden beim Anklicken wirklich nur die Taste hervorgehoben

Um die Gestaltung der Beispieldatenbank nach den Vorgaben der Apple Human Interface Guidelines weiter umzusetzen, widme ich mich im Folgenden einigen Navigationsgrafiken. Wie im ersten Teil beschrieben, habe ich im Kopfteil des Layouts eine Leiste eingefügt, in der sich Steuerelemente zur Navigation befinden. Nachdem ich das letzte Mal über die Tasten zum Wechseln zwischen den einzelnen Tabellen (❶) gesprochen habe, möchte ich nun noch ein paar weitere Elemente erwähnen. Da ich die FileMaker Statusleiste in den meisten meiner Lösungen dauerhaft ausgeblendet habe, benötige ich Tasten zum Wechseln zwischen den Datensätzen. Außerdem befinden sich in dieser Leiste Tasten, um von der Detailansicht zu einer Listenansicht zu gelangen (❷). Abschließend integriere ich rechts noch ein Suchfeld (❸) und ganz links das eigene Logo (❹):

Navigationsleiste



Die Elemente im Einzelnen

❶ Mit diesen Tasten navigiert man zwischen den Tabellen.

❷ Detail-, Listen- und Spaltenansicht. Bei den Symbolen für diese Tasten habe ich mich an den Grafiken orientiert, die im Mac OS X **Finder** verwendet werden.[1]

Im **Finder** gibt es insgesamt vier verschiedene Darstellungsformen für Objekte innerhalb eines Ordners:

 : Diese Grafik steht im Finder für die Darstellung des Objektes als Symbol. Das trifft zwar nicht hundertprozentig auf die Detailansicht zu, ist aber so zu werten, dass der Anwender in dieser Ansicht die größte Übersicht erhält.

 : Die Linien in dieser Grafik können als Liste interpretiert werden, weshalb sich dieses Symbol geradezu perfekt für das Wechseln zur Listenansicht der Datensätze eignet.

 : Dieses Symbol steht für den Wechsel zur sogenannten Spaltenansicht. Neben der Detail- und der Listenansicht biete ich dem Anwender auch diese Darstellungsform in den meisten meiner Lösungen an. Leider ist unsere Beispieldatei nicht besonders gut dafür geeignet, den Sinn der Spaltenansicht zu vermitteln. Sie ermöglicht das „Herantasten" an einen Datensatz und ist damit sehr hilfreich, wenn der Anwender nicht genau weiß, wonach er sucht. Man kann das Ganze auch als eine Art Filter beschreiben, der immer weiter einschränkt. Die Vorgehensweise erkläre ich in einem anderen Teil dieser Serie.

 : Diese sogenannte „Cover-Flow"-Darstellung ermöglicht es dem Anwender im Finder, zwischen den Inhalten eines Ordners zu blättern. Anwender von Apples **iTunes** und Besitzer von iPhone, iPod, iPad etc. kennen dieses Blättern ebenfalls. Da es mir aber bisher noch nicht gelungen ist, eine entsprechend coole Umsetzung für FileMaker zu entwickeln, findet die Taste bei mir keine Verwendung und es bleibt bei den drei oben aufgeführten Modi.

❸ Navigieren zwischen den Datensätzen. Würde man es ganz genau nehmen, müsste man hierfür in jeder Tabelle ein Formelfeld anlegen und entsprechende Grafiken positionieren. Da ich aber immer versuche, möglichst wenige Felder für Layoutzwecke anzulegen, verzichte ich auf die absolute optische Übereinstimmung des Pfeils in den Tasten und verwende stattdessen das Unicode-Zeichen 8227 „‣" aus der Schrift „Lucida Grande".

Diese Lösung birgt noch einen weiteren Vorteil: Da es sich um eine Schrift handelt, kann mit der „Bedingte Formatierung" gearbeitet werden, Befindet man sich nämlich im ersten bzw. letzten Datensatz, so ist ein Weiterblättern nach vorn bzw. hinten nicht mehr möglich. In diesem Fall ist die Taste inaktiv und somit grau. Um das zu erreichen, habe ich folgende „Bedingte Formatierung" verwendet:

Für die Zurück-Taste:

```
Hole (DatensatzPositionInErgebnismenge) -1
```

Für die Vorwärts-Taste:

```
Wenn (Hole (DatensatzPositionInErgebnismenge) =
     Hole (AnzahlGefundeneDatensätze); 1 ; 0)
```

Das Script für die Navigation habe ich folgendermaßen aufgebaut:

```
Navigation Datensätze
◆ Kommentar
    Text:  © 07.10.2010 E I N S A T Z Creative Production | Holger Darjus
           Steuert die Navigation durch die Datensätze
◆ Kommentar
    Text:  Vorwärts
◆ Wenn                                                              ▼
    Formel:  Hole (ScriptParameter) = "Vor"
◆ Kommentar
    Text:  Bei gedrückter ALT-Taste zum Letzten.
◆ Wenn                                                              ▼
    Formel:  Hole (SonderTastenGedrückt) = 8
◆ Gehe zu Datens./Abfrage/Seite
    Option:  Letzte(r)
◆ Sonst                                                             ◆
◆ Gehe zu Datens./Abfrage/Seite
    Option:  Nächste(r)
◆ Ende (wenn)                                                       ▲
◆ Kommentar
    Text:  Rückwärts
◆ Sonst, wenn                                                       ◆
    Formel:  Hole (ScriptParameter) = "Zurück"
◆ Kommentar
    Text:  Bei gedrückter ALT-Taste zum Ersten.
◆ Wenn                                                              ▼
    Formel:  Hole (SonderTastenGedrückt) = 8
◆ Gehe zu Datens./Abfrage/Seite
    Option:  Erste(r)
◆ Sonst Gehe zu Datens./Abfrage/Seite                               ◆
    Option:  Vorherige(r)
◆ Ende (wenn)                                                       ▲
◆ Ende (wenn)                                                       ▲
```

Im Vergleich zur Navigation in der FileMaker Statusleiste fällt vielleicht auf, dass „Gehe zu Datensatz-Nr." fehlt. Auf diese Funktion habe

[1] *Für reine Windows-Anwender: Der Finder ist Apples Entsprechung für den Explorer.*

ich verzichtet, da ich keinen vernünftigen Anwendungszweck dafür kenne.

Nun fehlen nur noch die Anzahl der gefundenen Datensätze, die Position des aktuellen Datensatzes und die Anzahl aller Datensätze in der Tabelle. Hierfür kann man entweder ein Feld in der Datenbank ablegen oder mit einer **$$**-Variablen arbeiten, was ich bevorzuge. Wie auch immer man sich entscheidet, die Formel sieht in meinen Lösungen meist so aus:

```
Hole (DatensatzPositionInErgebnismenge) & " / " &
Hole (AnzahlGefundeneDatensätze) & " / " &
Hole (AnzahlDatensätzeGesamt)
```

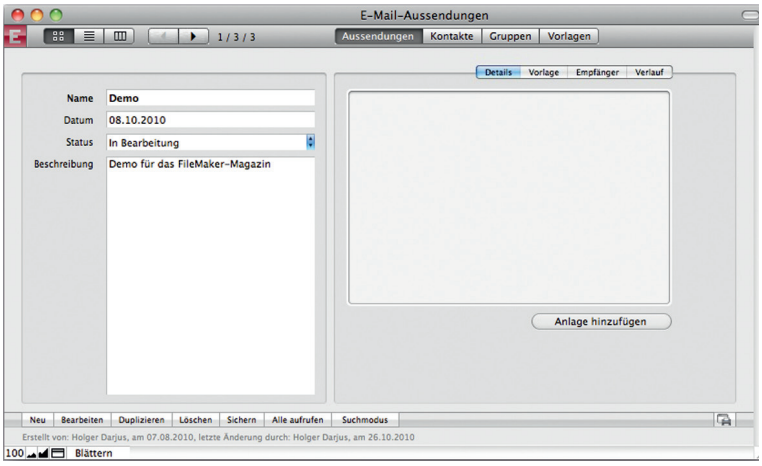
Damit ist unsere Kopfleiste fertig.

Was einen Kopf hat, sollte auch einen Fuß haben. Für den Fußteil habe ich nichts Brauchbares im Interface-Builder gefunden und daher etwas Eigenes kreiert. Im linken Teil meiner Fußleiste habe ich die Standard-Tasten zum Erstellen, Duplizieren, Bearbeiten, Löschen etc. untergebracht, die sich in allen meinen Lösungen immer an der gleichen Stelle befinden. Wenn eine dieser Tasten keine Funktion hat, weil zum Beispiel das Löschen von Datensätzen nicht erlaubt ist, wird sie grau dargestellt und dem Anwender so signalisiert, dass die Funktion nicht zur Verfügung steht. In einer QuickInfo kann nachgelesen werden, warum etwas nicht ausgeführt werden kann, oder was passiert, wenn die Taste gedrückt wird.

Im rechten Teil des Fußbereichs sind oft Tasten zu finden, die im Kontext mit der Anwendung und dem aufgerufenen Datensatz stehen. In unserer Beispiellösung ist das der „Sende-Knopf“. Im Original der Beispieldatei versteckt sich dieser Knopf rechts im oberen Viertel und ist zudem anklickbar, obwohl noch keinerlei Angaben zur Aussendung gemacht wurden. In der von mir überarbeiteten Version wird in dem Script „Senden“ zunächst abgefragt, ob alle relevanten Informationen eingetragen wurden. Sollte das nicht der Fall sein, verweigert der Knopf einfach den Dienst und ist grau. Unangenehm ist, dass die im Script definierten Bedingungen für einen Abbruch ein zweites Mal für die „Bedingte Formatierung“ der Taste gesetzt werden müssen. Bei sehr komplexen Bedingungen verwende ich daher ein Formelfeld, das vom Script und von der „Bedingten Formatierung“ gleichermaßen abgefragt wird.

Abschließend erstelle ich im untersten Bereich eine Zeile, in der die Meta-Informationen zum Datensatz angezeigt werden, wie z. B. Angaben darüber, von wem der Datensatz wann angelegt und geändert wurde.

Alles, was bisher besprochen wurde, sind Standards, die in den meisten meiner Lösungen vorkommen. Alles Weitere steht in Abhängigkeit zu der jeweiligen Aufgabe der Lösung. Unsere Beispieldatei sieht jetzt so aus:



Widmen wir uns nun einem Thema, das mich lange beschäftigt hat, bevor ich eine optimale Lösung gefunden habe: Es geht um die Zuordnung eines Wertes aus einer anderen Tabelle z. B. wenn in einer Auftragsbearbeitung ein Kunde oder eine Lieferadresse zugeordnet werden soll. Auch in unserer Beispieldatei gibt es einen solchen Vorgang: Unter dem Register „Empfänger“ hat der Anwender die Möglichkeit, der Aussendung einzelne Empfänger oder Gruppen zuzuordnen.

Wenn auf maximal 50 Auswahlmöglichkeiten zugegriffen wird, kann man sicher mit Wertelisten und Drop-down-Menüs arbeiten. Geht die Anzahl aber darüber hinaus oder gar in den Tausenderbereich, ist dieser Weg weder praktikabel noch schön.

Ich habe daher eine Routine entwickelt, die eine modale[2] Dialogbox öffnet, in der sich ein Browse-Feld und eine Liste aller Auswahlmöglichkeiten befinden. „Modal“ bedeutet, dass der Vorgang in irgendeiner Weise abgeschlossen sein muss, bevor der Anwender irgendetwas anderes tun kann. Damit werden falsche Zuordnungen vermieden, die z. B. dann passieren können, wenn ein Anwender bei seiner Arbeit mit der geöffneten Dialogbox unterbrochen wird. Wenn er dann – weil er nicht mehr genau weiß, was er zuletzt getan hat – den Datensatz wechselt, die geöffnete Dialogbox erst später wiederentdeckt und auf „OK“ klickt, ist nicht auszuschließen, dass er dabei eine unbeabsichtigte Zuordnung vorgenommen hat.

[2] *Dialogfenster haben meist eine geringere, oft feste Größe und erscheinen als Pop-up-Fenster vor dem Hauptfenster der Anwendung. Meist handelt es sich dabei um modale Fenster, das heißt, der Benutzer kann in den anderen Fenstern derselben Anwendung nicht weiterarbeiten, solange das Dialogfenster angezeigt wird. Quelle: de.wikipedia.org/wiki/Dialogfenster#Modale_und_nichtmodale_Fenster*

Um das zu unterbinden, erstellen wir eine modale Dialogbox, wozu wir ein Layout und ein Script benötigen. Das Script öffnet ein neues Fenster, wechselt zu dem entsprechenden Layout und sorgt dafür, dass der Anwender nichts anderes tun kann, als einen Kontakt auszuwählen und das Fenster mit „OK“ oder „Abbrechen“ zu schließen. Damit das funktioniert, wird in dem Script eine Schleife aufgerufen, die so lange läuft, bis eine der Bedingungen erfüllt ist.

Damit der Vorgang selbst durch das Schließen des Fensters über die Fenster-Buttons ( ) nicht unterbrochen werden kann, sollte das Script folgendermaßen aufgebaut sein:

```
Dialogbox
◆ Fenster fixieren
◆ Neues Fenster
    Name:    "Kontakt oder Gruppe zuordnen"
    Höhe:    500
    Breite:  350
    Oben:    30
◆ Gehe zu Layout
    Layout:  "Auswahl" (Aussendungen)
◆ Statusbereich ein-/ausblenden
    Option:  Fixieren; Ausblenden
◆ Fenster anpassen
    Option:  Alles anzeigen
◆ Zoomstufe setzen
    Option:  Fixieren
    Option:  100%
◆ AnwenderAbbruchZulassen setzen
    Option:  Aus
◆ Variable setzen
    Formel:  $$RUN
    Wert:    0
◆ Schleife (Anfang)                    ▼
◆ Scriptpause setzen
    Option:  Unbegrenzt
◆ Verlasse Schleife wenn               ◆
    Formel:  $$RUN > 0
◆ Schleife (Ende)                      ▲
```

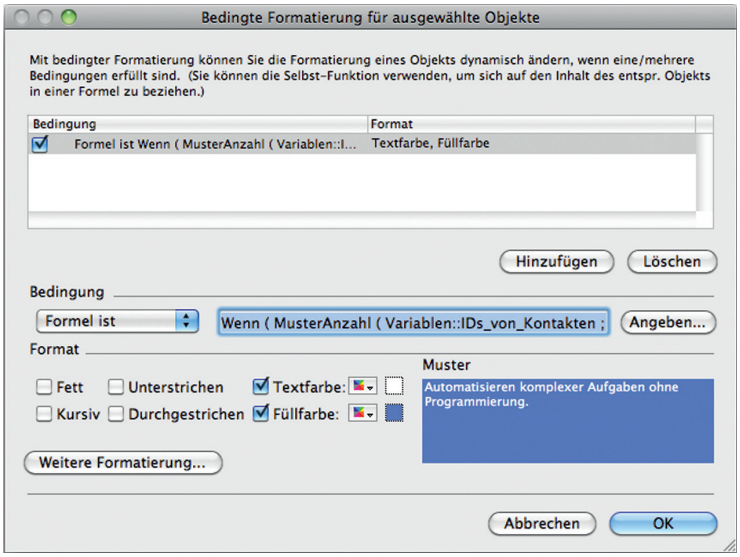
Da das Script in einer Endlos-Schleife läuft, kommt der Anwender aus der „Nummer“ nicht mehr raus, solange die Variable **$$RUN** keinen anderen Wert als „0“ hat. Um die Änderung dieses Werts kümmert sich das Script an einer anderen Stelle. Der Button „Abbrechen“ bewirkt beispielsweise, dass die Variable **$$RUN** den Wert „2“ bekommt. Was dann passiert, wird im zweiten Teil des Scripts abgefangen. Im „Abbrechen“-Fall werden die Datensatzänderungen verworfen und das Fenster wird geschlossen.

Was sonst noch in diesem Script geschieht, soll hier nicht weiter ausgeführt werden, denn es geht in dieser kleinen Serie ja nicht um ausgefeilte Scripts, sondern um die Gestaltung von Benutzerführung.

Die Anwendungsmöglichkeiten für einen solchen Vorgang sind vielfältig. Und da der Script-Aufbau und das Layout für das Fenster ohne große Anpassungen für andere Anwendungen übernommen werden können, lohnt sich der einmalige Gestaltungsaufwand allemal.

Kümmern wir uns also abschließend noch um die Gestaltung der Dialogbox. Zunächst erstelle ich im Interface-Builder das Grundgerüst für die Dialogbox und positioniere einen Screenshot davon auf einem neuen FileMaker-Layout.

Die Grundelemente, also die Tasten „OK“ und „Abbrechen“, das Browse-Feld und die Box für die angezeigten Inhalte, stammen bereits vom Screenshot. Fehlt nur noch das Portal, das ich mit dem Ausschnittwerkzeug aufziehe. Um die Höhe der Zeilen präzise zu bestimmen, begrenze ich die Anzahl der Zeilen zunächst auf eine und gebe später im Inspektor 18 Pixel als Zeilenhöhe ein. Ich setze die Option „Hintergrund abwechseln“ und wähle die Farbe „AlternateSelectedTextColor“.



Das Portal selbst bekommt die Füllfarbe „Weiß“ und Linienstärke „Keine“. Über dem Ganzen positioniere ich noch ein Element, das ich „Listenkopf“ nenne und in dem später die Überschriften für die Felder untergebracht werden. Anschließend gehe ich noch einmal in die Ausschnitteinstellungen, erweitere das Portal auf 31 Zeilen und versehe es mit einem vertikalen Rollbalken. Danach ziehe ich das Portal und den Listenkopf so weit auf, dass sie links und rechts an den Rand der Box stoßen und der Raum oben und unten gleichmäßig in der Box verteilt wird.

Ist das erledigt, positioniere ich die gewünschten Felder in dem Portal. In unserem Fall sind

das Vor- und Nachname des Kontakts sowie Firmenname. Nun müssen wir uns um die Beziehungen kümmern, damit die Datensätze auch angezeigt werden. Dafür habe ich zunächst eine Tabelle namens ***Variablen*** angelegt, in der sämtliche Grafiken und einige Hilfsfelder untergebracht sind. Zwei dieser Hilfsfelder dienen dem Browse-Feld: In eines dieser Felder kann der Anwender einen Suchbegriff eingeben, das andere ist ein Formelfeld. Solange das Eingabefeld leer ist, enthält dieses Formelfeld nur das Zeichen „*“, ansonsten wird es mit dem Inhalt des Eingabefeldes gefüllt:

```
Wenn (IstLeer (Browsefeld Eingabe); "*"; Browsefeld Eingabe)
```

In der Tabelle Kontakte habe ich ebenfalls ein Formelfeld angelegt, das zunächst immer das Zeichen „*“ enthält. Somit ist die Beziehung von der Tabelle Variablen zur Tabelle Kontakte für alle Datensätze gültig, solange der Anwender keine Eingabe in dem Browse-Feld vorgenommen hat.

Der nächste Teil ist etwas komplizierter: Dafür verwende ich eine Eigene Funktion, die Arnold Kegebein bis ins kleinste Detail ausgefeilt hat und in seinem Artikel auf Seite 13 genauer erläutert. Diese Funktion zerlegt einen Text so, dass z. B. aus „Holger Darjus“ Folgendes wird:

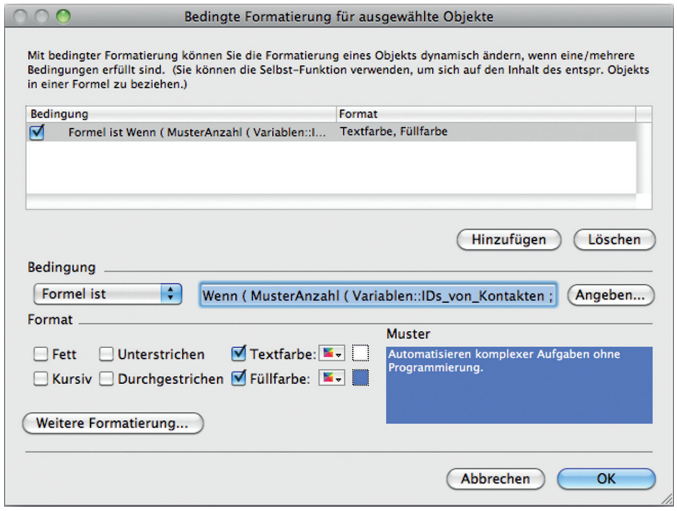
```
*
Holger Darjus
Holger Darju
Holger Darj
Holger Dar
Holger Da
Holger D
Holger
Holger
Holge
Holg
Hol
Ho
H
Darjus
Darju
Darj
Dar
Da
D
```

Das sieht zwar etwas kompliziert aus, macht die Sache für den Anwender aber ungemein komfortabel. Weist man dem Browse-Feld nämlich noch einen Script-Trigger[3] zu, der beim Verändern des Feldes ausgeführt wird, kann die Menge der angezeigten Datensätze in dem Portal bereits während der Eingabe eingeschränkt werden.

Fehlt nur noch die Auswahl der Kontakte. Dazu lege ich auf die Felder in der Ausschnittreihe eine Taste, die das Script „Dialogbox“ aufruft und den Parameter „Auswahl“ übergibt. Damit wird die ID des angeklickten Kontakts in das Feld **IDs_von_Kontakten** in der Tabelle ***Variablen*** geschrieben bzw. – wenn der Wert bereits vorhanden ist – entfernt.

```
Wenn (MusterAnzahl (Variablen::IDs_von_Kontakten;
Variablen | Kontakte::ID_KONTAKT & "¶") > 0;
Austauschen (Variablen::IDs_von_Kontakten;
Variablen | Kontakte::ID_KONTAKT & "¶"; "");
Variablen | Kontakte::ID_KONTAKT & "¶" & Variablen::IDs_von_Kontakten)
```

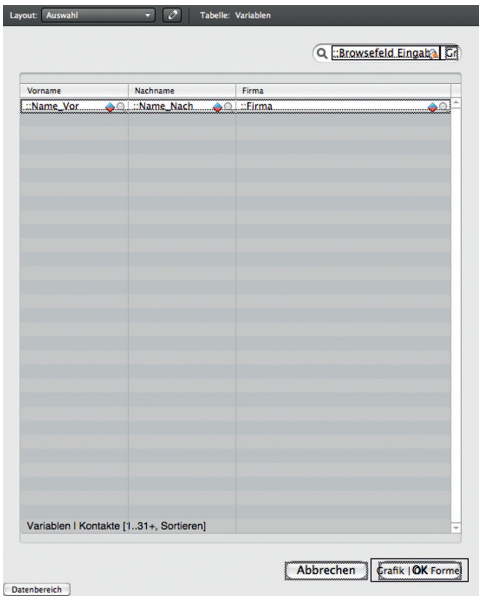
Damit der Anwender sehen kann, was er ausgewählt hat, bekommen die Felder in der Ausschnittreihe eine „Bedingte Formatierung“, die für eine Hervorhebung sorgt. Auch hierfür gibt es in den Apple Interface Guidelines vordefinierte Werte: Der Text wird weiß und der Hintergrund blau. Das Blau nennt sich „KeyboardFocusIndicatorColor“ und hat die RGB-Werte 88 150 219.

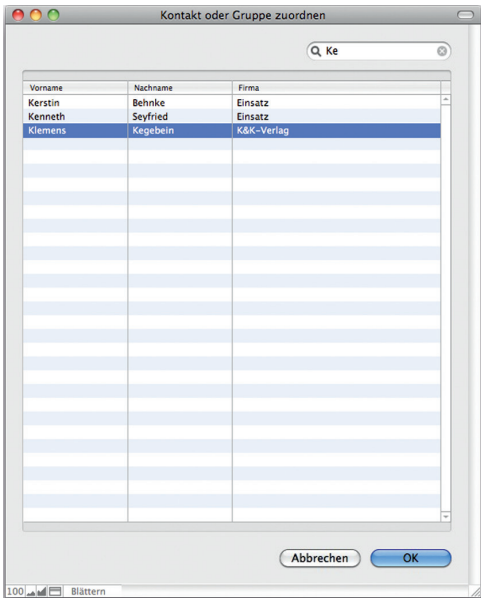


Wenn das Globalfeld **IDs_von_Kontakten** die ID des Kontakts in der Ausschnittreihe enthält, ist die Bedingung erfüllt und Text sowie Hintergrund färben sich entsprechend ein. Zugleich wird die Bedingung für das Formelfeld der „OK“-Taste „wahr“. Diese Taste wird nämlich erst dann aktiv, wenn mindestens ein Kontakt ausgewählt wurde. Dann verfärbt sich die Taste blau und signalisiert so dem Anwender, dass etwas passiert, wenn er sie anklickt.

Die fertig gestaltete Dialogbox sieht folgendermaßen aus: →

Script-Trigger Browsefeld
- ◆ *Kommentar*
  Text: *© 18.10.2010 EINSATZ Creative Production | Holger Darjus*
  Sorgt dafür, dass bei der Eingabe in das Browsefeld die Auswahl sofort eingeschränkt wird. — Wird durch einen Script-Trigger beim Verändern des Feldes ausgelöst.
- ◆ *Schreibe Änderung Datens./Abfrage*
  Option: Ohne Dialogfeld
- ◆ *Gehe zu Feld*
  Feld: Variablen | Variablen::Browsefeld Eingabe





Wer das Ganze noch abrunden möchte, sorgt dafür, dass bei geöffneter Dialogbox sämtliche Funktionen außer „Kopieren“ und „Einfügen“ im FileMaker Menü deaktiviert sind. Dafür erstellt man unter **Ablage → Verwalten → Angepasste Menüs …** ein eigenes Menü, das in der Rubrik „Bearbeiten“ nur diese beiden Punkte enthält.

Soweit unser kleiner Ausflug ins Reich der Dialogboxen. Im nächsten Teil wird es um Listen- und Spaltendarstellungen und um dynamische Layoutgrößen gehen. ♦

Haben wir noch Etwas zum Auffüllen?