

● FM MEETS MOZART · SALZBURG 2026

Neu in FileMaker 2026

Die wichtigsten Neuerungen für den Entwickleralltag.

Alexander Lemuth · ALESO SOLUTIONS

Montag, 14. September 2026 · 14:00–15:30 · Mozarteum Salzburg

WORUM ES GEHT

Kompakter Überblick über das Release

FileMaker 2026 ist da. Statt jede Zeile der Release Notes durchzugehen, ordne ich die Neuerungen für den Alltag ein – was lohnt sich für bestehende Lösungen, was für neue Projekte?

Zu jedem Thema gibt es eine **Live-Demo** aus einer vorbereiteten Datei.



Alexander Lemuth

Inhaber von ALESO SOLUTIONS · FileMaker-Entwickler mit Leidenschaft. Individuelle, skalierbare Lösungen und Schulungen.

AGENDA

Zwölf Neuerungen im Überblick



1 Layout Inspektor



2 Field Entry by Calculation



3 PDF-Funktionen



4 Window UUIDs



5 Custom Display Name



6 Configure Persistent Data



7 GetRecordIDsFromFoundSet



8 Script Workspace: Einklappen



9 Show Custom Dialog: Position



10 Export Field Contents (Server)



11 Leere Werte ans Ende sortieren



12 Ersetzen ohne Auto-Eingabe

Zu jedem Thema eine Live-Demo aus einer vorbereiteten Datei.

Klassisch oder Smart

Der Layout Inspektor bietet jetzt zwei Ansichten. **Der Smart Inspektor zeigt nur die Funktionen, die für das gewählte Element tatsächlich verfügbar sind** – alles Unnötige wird ausgeblendet.



Klassische Ansicht

Die gewohnte, vollständige Darstellung aller Bereiche – vertraut und immer gleich aufgebaut.



Smart Inspektor

Zeigt nur, was für das gewählte Element relevant ist. Weniger Suchen, weniger Ablenkung.



Zu finden unter: [Einstellungen](#) → [Schema](#) → [Adaptiven Bereich „Inspektor“](#) verwenden

Geschmackssache – mit Vor- und Nachteilen



Dafür

- Weniger visuelles Rauschen
- Schnellerer Zugriff auf das Relevante
- Gut für fokussiertes Arbeiten
- Angenehm für Einsteiger



Dagegen

- Elemente wandern je nach Auswahl
- Gewohnte Position nicht immer gleich
- Umgewöhnung für erfahrene Hände
- Nicht jeder mag adaptive Oberflächen

Mein Rat: einfach beide Ansichten ausprobieren – und die nehmen, mit der Sie schneller sind.

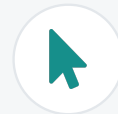
Field Entry by Calculation

Das Feldeingabeverhalten lässt sich pro Feld festlegen – **jeweils getrennt für Blätter- und Suchenmodus**. Neu: es kann per Formel gesteuert werden.



Bearbeiten

Feld kann geändert werden.



Nur auswählen

Markieren und kopieren, aber nicht ändern.



Nur anzeigen

Feld kann nicht betreten werden.



Durch Formel festgelegt

Eine Formel bestimmt das Verhalten zur Laufzeit.

Der vierte Modus ist der spannende – den schauen wir uns genauer an.

Drei Zustände über den Rückgabewert

1	Bearbeiten	Feld kann geändert werden
2	Nur auswählen	markieren / kopieren, nicht ändern
0 (o. a.)	Nur anzeigen	Feld kann nicht betreten werden

Der Rückgabewert der Formel entscheidet – Feld für Feld, zur Laufzeit.

Beispiel: Status eines Auftrags steuert die Bearbeitbarkeit

```
Case (  
  AUFTRÄGE::Status = "OFFEN" ; 1 ;  
  AUFTRÄGE::Status = "STORNIERT" ; 2 ;  
  AUFTRÄGE::Status = "ABGESCHLOSSEN" ; 0  
)
```

Was das bewirkt

OFFEN

wird bearbeitbar

STORNIERT

nur auswählbar

ABGESCHLOSSEN

nur anzeigbar

Keine Trigger, keine Umwege – die Logik sitzt direkt am Feld.

PDFs im Speicher aufbauen

Neue Script-Schritte erlauben es, PDFs **komplett im Speicher zu erzeugen und zusammenzusetzen** – erst am Ende wird gespeichert oder ausgegeben.



Create PDF

Startet ein neues, leeres PDF im Speicher.



Open PDF

Öffnet ein bestehendes PDF im Speicher.



Append PDF

Fügt dem offenen PDF weitere Seiten hinzu.



Close PDF

Speichert das PDF – in Pfad, Container oder Variable.



Cancel PDF

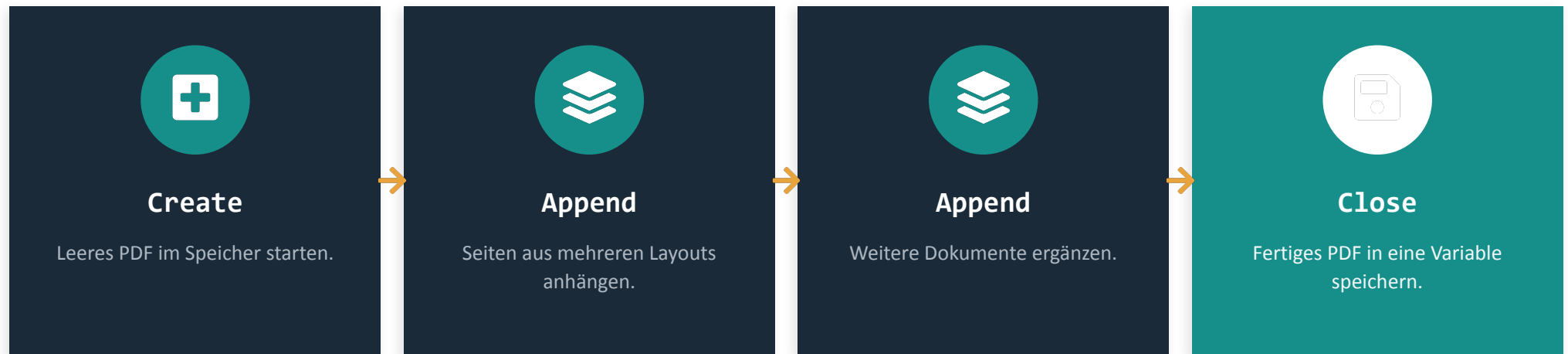
Verwirft das offene PDF ohne zu speichern.



Print PDF

Druckt ein PDF aus Pfad, Container oder Variable.

Ein Sammel-PDF in vier Schritten



Quelle & Ziel: PDFs kommen aus einem **Containerfeld (Source)** oder einem **Pfad (File)** – und lassen sich ebenso wieder in Container oder Pfad ausgeben.

Abschluss: **Close** speichert, **Cancel** verwirft, **Print** gibt direkt auf den Drucker aus.

Fenster eindeutig ansprechen



Fenstername

Für den Menschen – lesbar, aber nicht eindeutig. Zwei Fenster können denselben Namen tragen.



Window UUID

Für den Entwickler – eindeutig pro Fenster. Damit spricht ein Script genau das richtige Fenster an.

Warum das nützlich ist

- Kein Verwechseln
gleichnamiger Fenster mehr – Scripts treffen zuverlässig das gemeinte Fenster.
- Robuster bei mehreren offenen Fenstern
– etwa in komplexen, mehrfenstrigen Lösungen.

Ein Anzeigename, getrennt vom Feldnamen

Der echte Feldname bleibt unverändert im Schema. **FileMaker** kann aber je nach Oberfläche einen anderen, verständlicheren Namen anzeigen.

Echter Feldname

bleibt im Schema

```
z_Erstellungs-  
zeitstempel
```



Angezeigt - je nach Kontext



beim Export

Erstellungszeitpunkt



beim Sortieren

Erstellt am



in der Tabellenansicht

Zeitstempel Erstellung

Der Aha-Punkt: nicht einfach ein Alias - sondern ein eigener Anzeigename pro Kontext.

THEMA 5 · DEFINITION

Ein Feld, vier Kontexte

Die Anzeigenamen werden als JSON hinterlegt – ein Schlüssel je Kontext:

```
JSONSetElement ( "{}" ;  
  [ "fm_common"      ; "" ; JSONString ] ;  
  [ "fm_export"      ; "Kunden-ID" ; ... ] ;  
  [ "fm_sort"        ; "Kundennummer" ; ... ] ;  
  [ "fm_table_view"  ; "Kunden-Nr." ; ... ]  
)
```

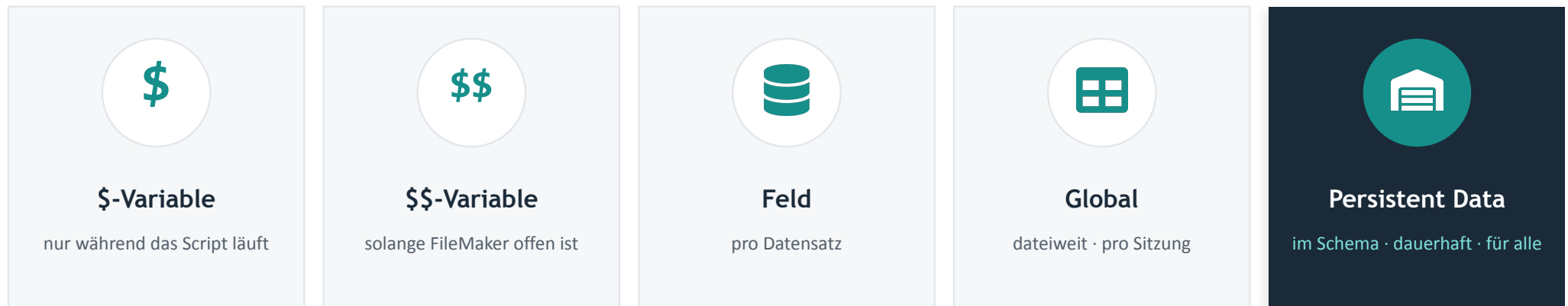
PERSONEN::_pk_personen_id

fm_common	(<i>leer</i> → Feldname)
fm_export	Kunden-ID
fm_sort	Kundennummer
fm_table_view	Kunden-Nr.

Zu fm_common: der allgemeine Anzeigenname für alle Kontexte ohne eigene Angabe. **Ist er leer, fällt FileMaker auf den echten Feldnamen zurück.**

Eine neue Speicherebene

Weder Variable noch Feld noch Global – sondern ein **Key-Value-Speicher direkt im Schema der Datei**. Er bleibt über Sitzungen hinweg erhalten und gilt für alle Benutzer.



Der Unterschied: adressierbar aus jedem Script und jeder Berechnung – ohne Datensatz, ohne Tabellenvorkommen, ohne Global-Ballast.

Ein Eintrag - und drei neue Werkzeuge

Jeder Eintrag besteht aus drei Teilen:

Name

wie der Wert heißt

Instance ID

optionaler Namensraum zum Gruppieren

Value

beliebiger FileMaker-Datentyp

Drei neue Bausteine steuern ihn:



Configure Persistent Data

Script-Schritt: Eintrag setzen, ändern oder löschen.



GetPersistentData

Funktion: einen Wert wieder auslesen.



ListPersistentDataIDs

Funktion: alle Instanzen zu einem Namen finden.

App-Version global, Config pro Modul

```
# Globaler Wert (kein Instance ID = dateiweit)
Configure Persistent Data
  [ AppVersion ; "" ; "2.4.1" ]

# Modul-Wert (Instance ID = Namensraum)
Configure Persistent Data
  [ Config ; "ModulRechnungen" ; $json ]

# Auslesen
GetPersistentData ( "AppVersion" ; "" )
```

Bleibt erhalten

Überlebt Kopie, Klon und komprimierte Kopie.
Auf dem Server sogar geräteübergreifend lesbar und schreibbar.

Wofür man es nutzt

- ✓ App-Version der eigenen Lösung
- ✓ JS-Bibliotheken für Web Viewer
- ✓ Feste Prompts für KI-Modelle
- ✓ Konfiguration pro Add-on-Instanz



Kein Ort für Secrets

Werte stehen im XML-Export im Klartext – keine API-Keys oder Passwörter hier ablegen.

Datensatz-IDs abgreifen

Die Funktion liefert die Record-IDs der aktuellen Fundmenge – als Liste oder JSON-Array. **Mit optionalem zweiten Parameter auch die IDs verknüpfter Datensätze.**



Seit Version 22

IDs der Fundmenge oder einer Tabellenbeziehung – also verknüpfte Datensätze über ein Tabellenvorkommen.

NEU 2026



Jetzt auch aus dem Portal

IDs direkt aus einem Portal – gefiltert und sortiert genau nach dessen Einstellungen. Über den Objektnamen des Portals.

Der Unterschied: Ein Portal kann gefiltert und anders sortiert sein als die Beziehung – genau diese sichtbare Auswahl bekommen Sie jetzt.

Aus dem Portal zur Ergebnisliste

Aufträge aus dem Portal einer Person greifen – und daraus eine echte Fundmenge bauen:

```
# 1) IDs aus dem Portal holen
Set Variable [ $RecordIDs ;
  GetRecordIDsFromFoundSet ( 0 ;
    "PERSONEN_Aufträge_Portal" ) ]

# 2) Fundmenge daraus erzeugen
Go to List of Records [
  $RecordIDs ; Layout: "AUFRÄGE_Liste" ]
```

Was passiert



Portal liefert – die IDs seiner sichtbaren Zeilen



Go to List – erzeugt daraus eine Fundmenge



Ergebnis – echte Liste im Ziellayout

Der Objektname zählt: „PERSONEN_Aufträge_Portal“ ist der Objektname des Portals – nicht der Beziehungsname. So kommt genau die gefilterte, sichtbare Auswahl.

Fünf Ausgabeformate

Der erste Parameter bestimmt, wie die IDs zurückkommen – als Konstante oder Zahl angebbbar:

type	Rückgabe	Beispiel
ValueNumber (0)	Liste, durch Absätze getrennt	1 ¶ 5 ¶ 21 ¶ 22 ¶ 23 ¶ 7
JSONString (1)	JSON-Array als Strings	["1", "5", "21", "22", "23", "7"]
JSONNumber (2)	JSON-Array als Zahlen	[1, 5, 21, 22, 23, 7]
ValueNumberRanges (3)	Liste mit Bereichen	1 ¶ 5 ¶ 21-23 ¶ 7
JSONStringRanges (4)	JSON-Array mit Bereichen	["1", "5", "21-23", "7"]

Bereiche (3/4): aufeinanderfolgende IDs werden zu 21-23 zusammengefasst – das hält die Rückgabe kompakt.

Kommentare & Schritte einklappen

Lange Scripts bleiben übersichtlich: **Kommentarblöcke und deaktivierte Schritte lassen sich zusammenklappen** – man sieht auf einen Blick die aktive Logik, ohne den Kontext zu verlieren.



Kommentare einklappen

Ausführliche Kommentarblöcke lassen sich zuklappen – die Dokumentation bleibt erhalten, stört aber nicht beim Lesen.



Deaktivierte Schritte einklappen

Vorübergehend abgeschaltete Schritte verschwinden optisch, bleiben aber im Script – ideal beim Testen und Debuggen.

Weniger Scrollen, klarerer Blick auf das Wesentliche – gerade bei großen Scripts ein spürbarer Gewinn.

Größe und Position frei bestimmen

Der Dialog erscheint nicht mehr nur zentriert. **Größe und Position lassen sich jetzt gezielt festlegen** – als fester Wert in Punkt oder per Formel berechnet.



Size

Height · **Width** – die Abmessungen des Dialogfensters.



Position

From Top · **From Left** – wo das Fenster auf dem Bildschirm sitzt.

Per Formel (fx): Größe und Position dynamisch berechnen – etwa abhängig von Bildschirmgröße oder Inhalt des Dialogs.

Feldinhalt exportieren - jetzt serverseitig

Der Script-Schritt „**Feldinhalt exportieren**“ läuft jetzt auch **serverseitig** – in Scripts auf FileMaker Server sowie über die Data API und die OData API.



FileMaker Server

in serverseitigen Scripts



Data API

bei REST-Zugriffen



OData API

bei OData-Zugriffen

Was das bringt: Exporte laufen ohne geöffneten Client durch – etwa geplante Ausgaben, API-Downloads oder automatisierte Dateieexporte direkt auf dem Server.

Leere Werte ans Ende sortieren

Beim Sortieren gibt es jetzt die Option, **leere Felder ans Ende zu stellen** – statt wie bisher an den Anfang.

Bisher

```
(leer)
(leer)
Anna
Müller
Schmidt
```

Neu: leere ans Ende

```
Anna
Müller
Schmidt
(leer)
(leer)
```

Gerade in Berichten und Listen wirkt das aufgeräumter – die befüllten Datensätze stehen oben.

Ersetzen ohne Auto-Eingabe

Beim „**Feldinhalt ersetzen**“ lässt sich die automatische Eingabe jetzt überspringen – der Wert wird gesetzt, ohne Auto-Eingabe-Optionen erneut auszulösen.



Auto-Eingabe übersprungen

Änderungszeitstempel, berechnete Werte oder Seriennummern werden nicht neu ausgelöst – die Daten bleiben unangetastet.



Wann das hilft

- Massenkorrekturen ohne Zeitstempel-Update
- Daten migrieren, Historie erhalten
- Berechnungen bewusst nicht neu anstoßen

Auch beim einzelnen Feld: Wer gezielt nur ein Feld neu schreibt, löst die Auto-Eingabe der übrigen Felder nicht aus – der Klassiker: korrigieren, ohne dass „geändert am“ auf heute springt.

ZUSAMMENFASSUNG

Zwölf Neuerungen, die den Alltag verbessern

- ✓ 1. Layout Inspektor
- ✓ 2. Field Entry by Calculation
- ✓ 3. PDF-Funktionen
- ✓ 4. Window UUIDs
- ✓ 5. Custom Display Name
- ✓ 6. Configure Persistent Data
- ✓ 7. GetRecordIDsFromFoundSet
- ✓ 8. Script Workspace: Einklappen
- ✓ 9. Show Custom Dialog: Position
- ✓ 10. Export Field Contents (Server)
- ✓ 11. Leere Werte ans Ende sortieren
- ✓ 12. Ersetzen ohne Auto-Eingabe

Viele kleine Verbesserungen, die sich im Alltag summieren - zu jeder gibt es eine Beispieldatei.



Fragen & Austausch

Welche Neuerung hilft Ihnen im Alltag am meisten?

Alexander Lemuth · ALESO SOLUTIONS · www.aleso-solutions.com

Vielen Dank an FM meets Mozart und das Mozarteum Salzburg.