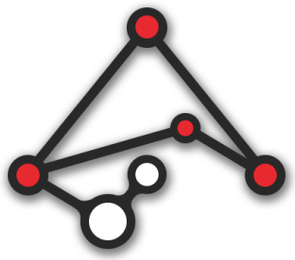
FileMaker Konferenz

FileMaker trifft Node.js

JavaScript-basierte Lösungen nahtlos im WebViewer einsetzen

Dr. Adam G. Augustin



www.agametis.de

Wer bin ich?

- Selbständiger FileMaker Entwickler im Raum München
- Beratung und Entwicklung seit über 15 Jahren
- Entwicklung von kundenspezifischen Datenbanken sowie Betreuung und Weiterentwicklung bestehender Lösungen
- Web- und App-Entwicklung
- FileMaker zertifiziert
- Zahlreiche Vorträge auf der FMK und dotfmp
- Mehr zu meinen Projekten mit Arbeitsbeispielen auf www.agametis.de und https://github.com/agametis
- FileMaker Magazin Award 2024 Gewinner

Inhalt

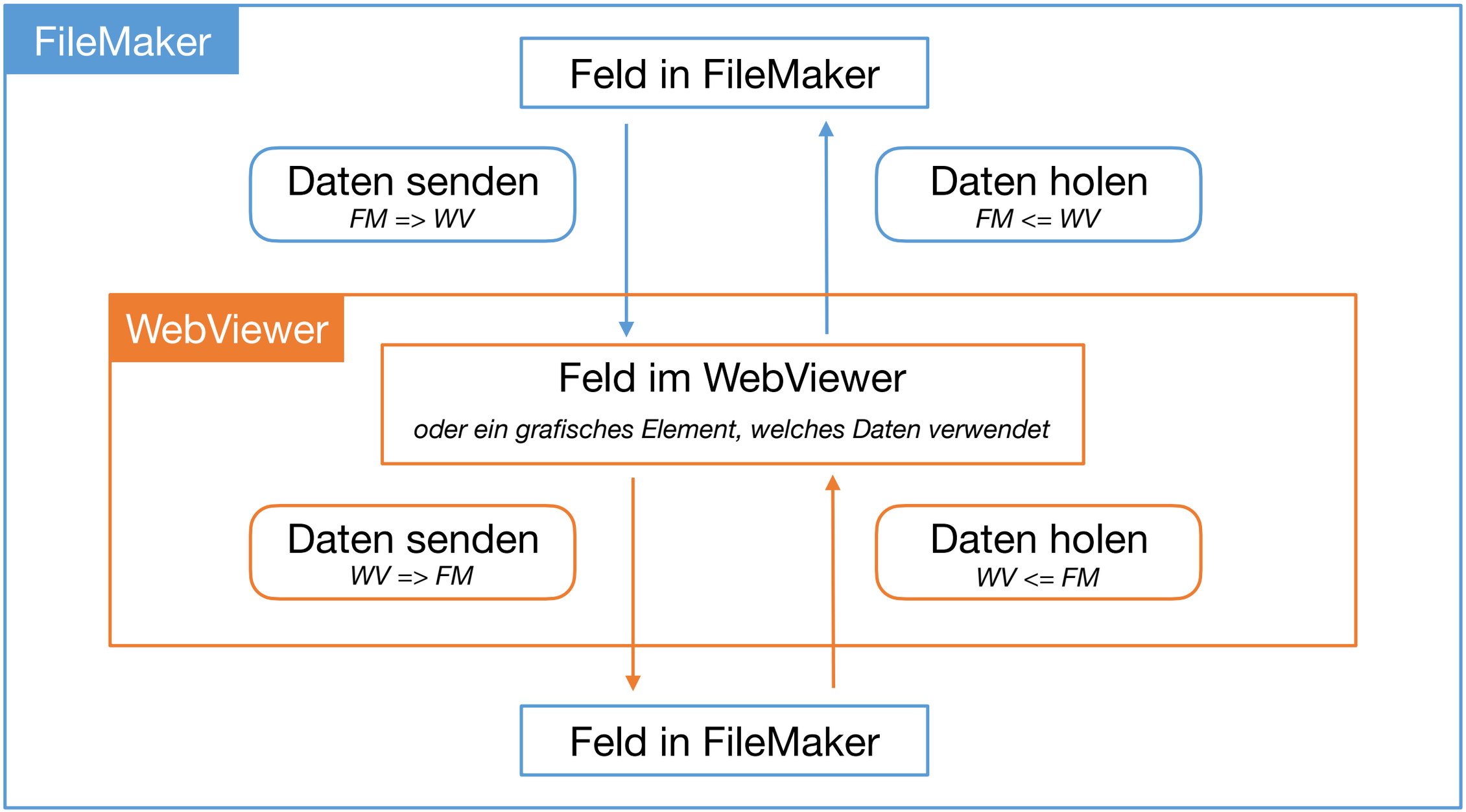
- Entwicklung mit Node.js und Vite.js
- Projektstruktur verstehen
- Deployment in FileMaker
- Tipps
- FAQ

Demo

https://github.com/agametis/fm-starter-vite

https://www.npmjs.com/package/@agametis/create-app-for-fm

Datenflüsse zw. FileMaker und WebViewer



Senden und Holen können z.B. durch Knöpfe (sowohl in FM als auch in WV) oder Interaktionen mit grafischen Elementen im WV ausgelöst werden.

Datenflüsse in der FM-Starter Datei

https://github.com/agametis/fm-starter-vite

Befehle für den Datentransfer

- In FileMaker Skripten:
  - Skriptschritt Perform JavaScript in Web Viewer.
  - Im Skriptschritt Generate Response from Model kann man auch Perform JavaScript in Web Viewer nutzen.
  - Skriptschritt Perform SQL Query by NL ermöglicht in manchen Konfigurationen auch Perform JavaScript in Web Viewer zu nutzen.

ab FileMaker 2025

- Im WebViewer:
  - Mit der JS-Funktion FileMaker.PerformScriptWithOption()
  - Aus Kompatibilitätsgründen ist auch die JS-Funktion FileMaker.PerformScript() verfügbar.

Direkte Links zu CLI App und Starter

https://ag.amet.is/app

https://www.npmjs.com/package/@agametis/create-app-for-fm

https://ag.amet.is/starter

https://github.com/agametis/fm-starter-vite

Entwicklungsumgebung

Ziele der Entwicklungsumgebung

- Entwicklung soll möglichst bequem und zeitsparend ablaufen.
- Es soll eine einfache Einbindung beliebiger Bibliotheken erlauben.
- Unser Code soll ohne Weiteres in WebDirect funktionieren (und damit funktioniert es auch in FMP und FMGo).
- Unser Code soll nach Möglichkeit keine “Cross-Origin Resource Sharing”-Probleme (CORS) haben (sonst wird ein Webserver nötig sein).

Node.js mit Vite.js als Basis

- Node.js (als JavaScript Runtime, https://nodejs.org/)
  - effiziente Paketverwaltung
  - Zugriff auf moderne JavaScript-Bibliotheken
  - schnelles Testen und Debuggen
- Vite.js (als Bundler, https://vite.dev/)
  - lokaler Entwicklungsserver
  - bietet schnelles “Hot-Reloading”
  - einfache und minimalistische Konfiguration
  - automatische Optimierung und Bündelung

Vorteile der Entwicklungsumgebung

- frei von Bibliotheken aus Content Delivery Networks
  - Bibliotheken wird nicht zur Laufzeit aus dem Internet geladen
  - Problem offenbart am “Polyfill Attack” im Februar 2024 (https://www.youtube.com/watch?v=F4h_N1Cz5dE oder https://www.youtube.com/watch?v=rXFSkDWVE6Y)
    - Erkenntnis: Dateien können relativ einfach manipuliert werden
- Es wird ein zentrales Verzeichnis für JavaScript-Pakete (https://www.npmjs.com/) benutzt. Und trotzdem:
  - Bibliotheken sind lokal im Projekt
  - feste Versionen, Sicherheitschecks und Audits möglich
  - Aber auch npmjs ist vor Angriffen nicht gefeit: https://www.youtube.com/watch?v=QVqlx-Y8s-s (Angriff vom 08.09.2025)
- Die eigene Programmierung ist einfach in FileMaker aktualisierbar

Voraussetzungen

- Lokal installierte Node.js-Umgebung (https://nodejs.org/), die im Terminal mit dem Befehl “node” verwendet werden kann
- der Kommandozeilen Package Manager “npm” und “npx” werden automatisch mitinstalliert
- Ein moderner Editor. Zu empfehlen wären:
    - Microsoft Visual Studio Code: https://code.visualstudio.com/ oder VSCodium: https://vscodium.com/
    - Windsurf: https://windsurf.com/ oder Cursor: https://cursor.com/, die eine integrierte KI-Unterstützung bieten.
- FileMaker ab Version 19.4
    - Skriptschritt “Perform JavaScript in Web Viewer”
    - JavaScript-Funktion "FileMaker.PerformScriptWithOption()” im WebViewer
- Optional, aber sehr zu empfehlen: lokal installiertes Git-Versionierungssystem (auf dem Mac enthalten, für Windows verfügbar unter https://git-scm.com/downloads).

Was macht ein Node.js Projekt aus?

Struktur eines Node.js-Projekts

- package.json: Projektdefinition, Skripte, Abhängigkeiten
- node_modules: Verzeichnis mit lokal installierten Bibliotheken
- vite.config.js: Projektkonfiguration
- index.html und main.js: Einstiegspunkt im Starterprojekt
- dist: Verzeichnis mit der Produktionsdatei (kompiliert für den WebViewer)

package.json

- definiert und organisiert ein Node.js-Projekt
- enthält alle relevanten Einstellungen eines Projektes und der Entwicklungsumgebung
  - Versionsinformationen
  - Packet-Abhängigkeiten
  - Skripte
- kann mit npm init erzeugt werden

Praktische npm-Befehle

- Allgemein:
  - npm init: Projekt initialisieren
  - npm install <packetname>: installiert ein Packet
  - npm install: installiert alle Abhängigkeiten
  - npm run <scriptname>: Skript ausführen (node-Skripte)
- Vite.js spezifische Skripte:
  - npm run dev: Entwicklungsserver starten
  - npm run build: Produktivversion erzeugen
- Zusätzlich in fm-starter-vite:
  - npm run deploy-to-fm: Überträgt den Code direkt zu FileMaker

Übertragung zu FileMaker

- Produktivversion erstellen (npm run build)
- Inhalt der index.html-Datei aus dem dist-Verzeichnis in ein Feld in FileMaker manuell kopieren.

Oder:

- Mit npm run deploy-to-fm den Code automatisch im richtigen Feld aktualisieren. Dabei wird der WebViewer gleich auch automatisch aktualisiert (mit dem bereitgestellten Skript).

Aktualisierung mit “deploy-to-fm”

- “Ein-Klick-Aktualisierung” des Codes in FileMaker
  - Aus der Node.js Umgebung heraus wird mit Hilfe des fmp-Protokolls ein FileMaker-Skript getriggert.
  - Das FileMaker-Skript lädt die Daten in ein Feld.
  - Das Feld enthält den gesamten Code für den WebViewer
    - nur eine einzige HTML-Seite, die den gesamten HTML, JS und CSS Code enthält

Best Practices & Tipps

Best Practices & Tipps

- Trennung von Logik und UI sehr einfach umsetzbar
  - Der Code kann in verschiedenen Dateien und Verzeichnissen organisiert
- Verwendung von Git für die Versionskontrolle zum “speichern” von Zwischenständen
- Verwendung von KI als Assistent für die JavaScript-Entwicklung
- Weitere Infos mit Details zu Codeteilen in meinen Präsentationen von der FMK2023 in Basel und FMK2024 in Malbun
  - https://github.com/agametis/fmk2023_demos
  - https://github.com/agametis/fmk2024_demos

Besonderheiten bei WebDirect

- Der Code (HTML, JS und CSS) sollte als String geladen werden.
- Referenz im WebViewer sollte auf eine Feld verweisen
  - keine Referenz auf eine Variable oder ein Verzeichnis auf der Platte
- Mit der FM-Funktion Base64Encode wird der Code in das Base64-Format umgewandelt …
- … und an das Prefix “data:text/html;base64,” angehängt (das Komma am Ende ist wichtig!).
- Die Base64-Umwandlung hat sich als hilfreich für die Funktionalität in WebDirect herausgestellt, um z.B. Probleme mit Sonderzeichen zu umgehen, aber …
- … ein Nachteil ist, dass die Daten um ca. 33% größer sind.

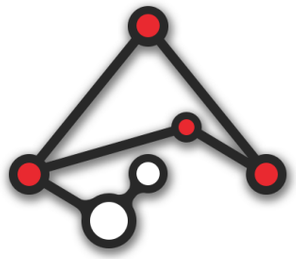
KI - dein Freund und Helfer

- JavaScript ist eine Programmiersprache, bei der man in der Entwicklung von der KI Unterstützung sehr stark profitieren kann
- Es geht sehr, sehr viel, ohne dass man selber eine einzige Zeile Code schreiben muss
- Integration mit FileMaker ist noch eingeschränkt
- Ein MCP-Server könnte in dieser Situation helfen
- In der Entwicklung (Veröffentlichung noch offen): “filemaker-wv” als MCP-Server ist eine Möglichkeit, sich Unterstützung zu holen:
  - https://www.npmjs.com/package/@agametis/filemaker-webviewer-mcp

FAQ

Vielen Dank

Bis morgen im Workshop



info@agametis.de

Vielen Dank unseren Sponsoren und Konferenz-Partnern

WORKSHOP zu
FileMaker trifft Node.js

Weitere Infos auf
https://github.com/agametis/fmk2025_demos

https://www.npmjs.com/package/@agametis/create-app-for-fm

https://github.com/agametis/fm-starter-vite

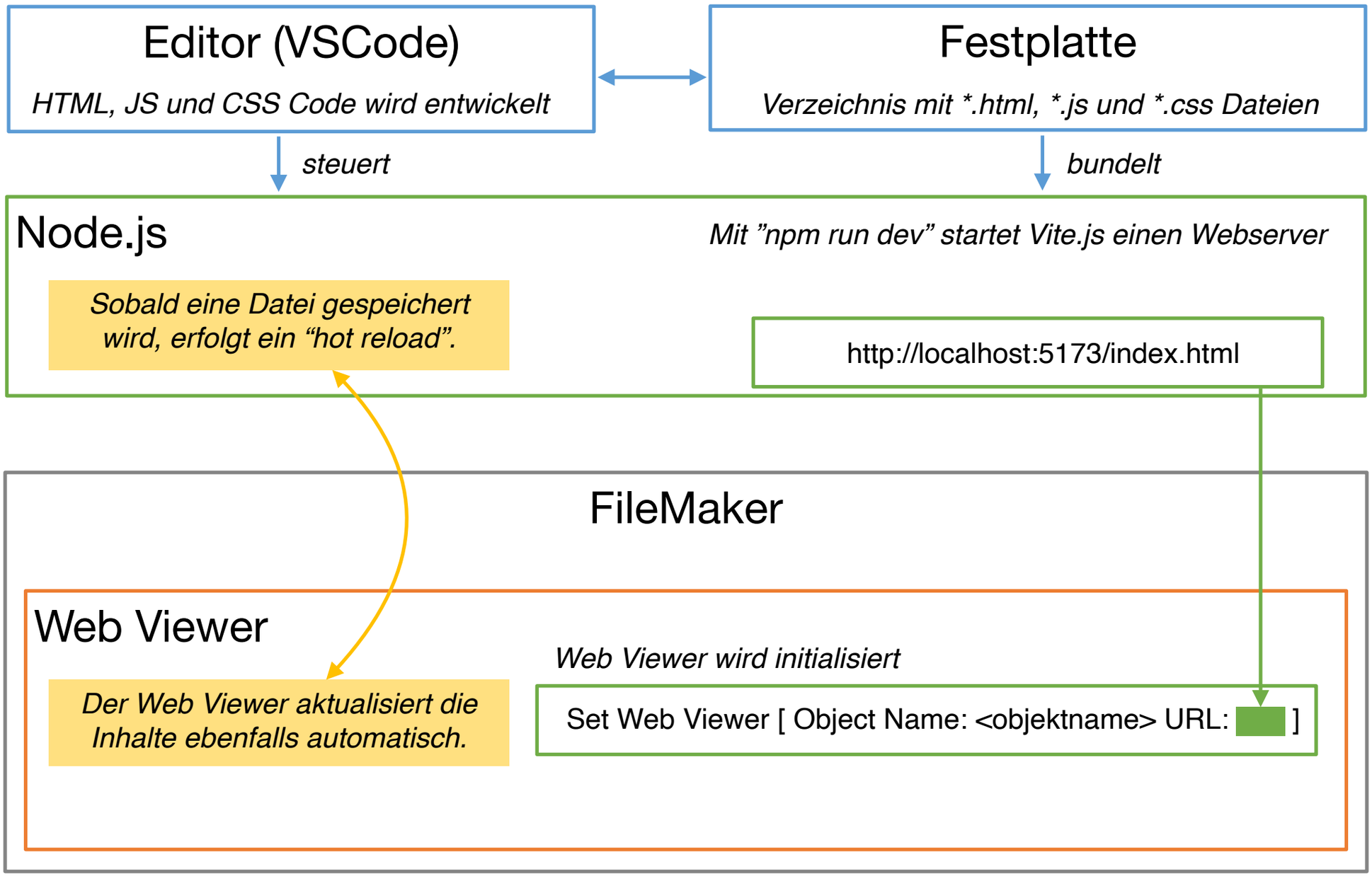
Voraussetzungen

- FileMaker ab Version 19.4
- Node.js und Git installiert
- IDE (Editor): VSCode, Windsurf oder Cursor (alle drei haben eine ausgewachsene KI-Unterstützung an Board)

Nützliche Extensions für VSCode & Co

- Console Ninja (WallabyJs)
- File Utils (sleistner)
- Open in External App (YuTengjing)
- Markdown Preview Enhanced (shd101wyy) oder Markdown All in One (yzhang)
- Live Server (ritwickdey)
- German - Code Spell Checker (Street Side Software)
- German Language Pack for Visual Studio Code (MS-CEINTL)
- GitHub Copilot in VSCode (die anderen Editoren haben ihre eigenen integrierten AI-Assistenten)
- “Roo Code” oder “Cline” als zusätzliche KI-Assistenten möglich (ganz neu ist jetzt auch “Kilo Code AI Agent” möglich)

Tooling (während der Entwicklung)



FMGofer als “Ersatz” für *FileMaker*

- FMGofer ist eine Bibliothek, die sich um alles kümmert, wenn es darum geht, mit FileMaker zu kommunizieren:
  - Es ist ein Wrapper für das “FileMaker”-Element im WebViewer
  - https://github.com/jwillinghalpern/fm-gofer
    - Prüft selbsttätig, ob das “FileMaker”-Element im WebViewer vorhanden ist
    - Ermöglicht eine asynchrone Ausführung des Codes
    - Bietet ein paar weitere Zusatzfunktionen out-of-the-box wie Timeout und JSON-Parsing
- In einem Node.js-Projekt sehr einfach einsetzbar

Legen wir los

- Sind Node.js und Git installiert?
- Ist VSCode oder eine äquivalente IDE installiert?
- Wenn Git installiert ist, kann das folgende CLI-Tool zum Laden des Startprojektes für einen einfachen Einstieg genutzt werden:
  - https://www.npmjs.com/package/@agametis/create-app-for-fm
- Ohne Git muss das Startprojekt manuell von GitHub heruntergeladen werden:
  - https://github.com/agametis/fm-starter-vite
- Mit Hilfe der README.md geht es dann im Editor und FileMaker weiter…

Projekt Starten mit CLI

- CLI-Tool ist zu finden unter:
  - https://www.npmjs.com/package/@agametis/create-app-for-fm
- Im Terminal:
  - Für eine temporäre Verwendung:
    - “npx @agametis/create-app-for-fm” ausführen
  - Für eine permanente Installation:
    - “npm i -g @agametis/create-app-for-fm” ausführen
  - “create-app-for-fm **create**” ausführen und den Anweisungen folgen

Workshop Themen

- Git zur “Versionskontrolle” vor allem im Zusammenhang mit KI
- Funktionen: initWebViewer, initialisiereWebViewer, datenVonFMHolen und datenAnFMSenden
- Bibliothek: FMGofer
- Der window-Kontext und Funktionen ohne Referenzen
- Reorganisation der Logik (Refactoring)
- Skript: deploy-to-fm