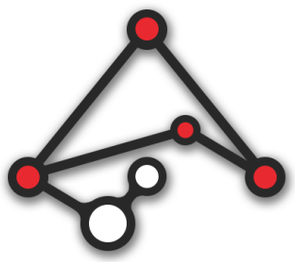
FileMaker Konferenz

# JavaScript in FileMaker einsetzen

## JavaScript Track 2/4 (Gantt-Chart)

Dr. Adam G. Augustin



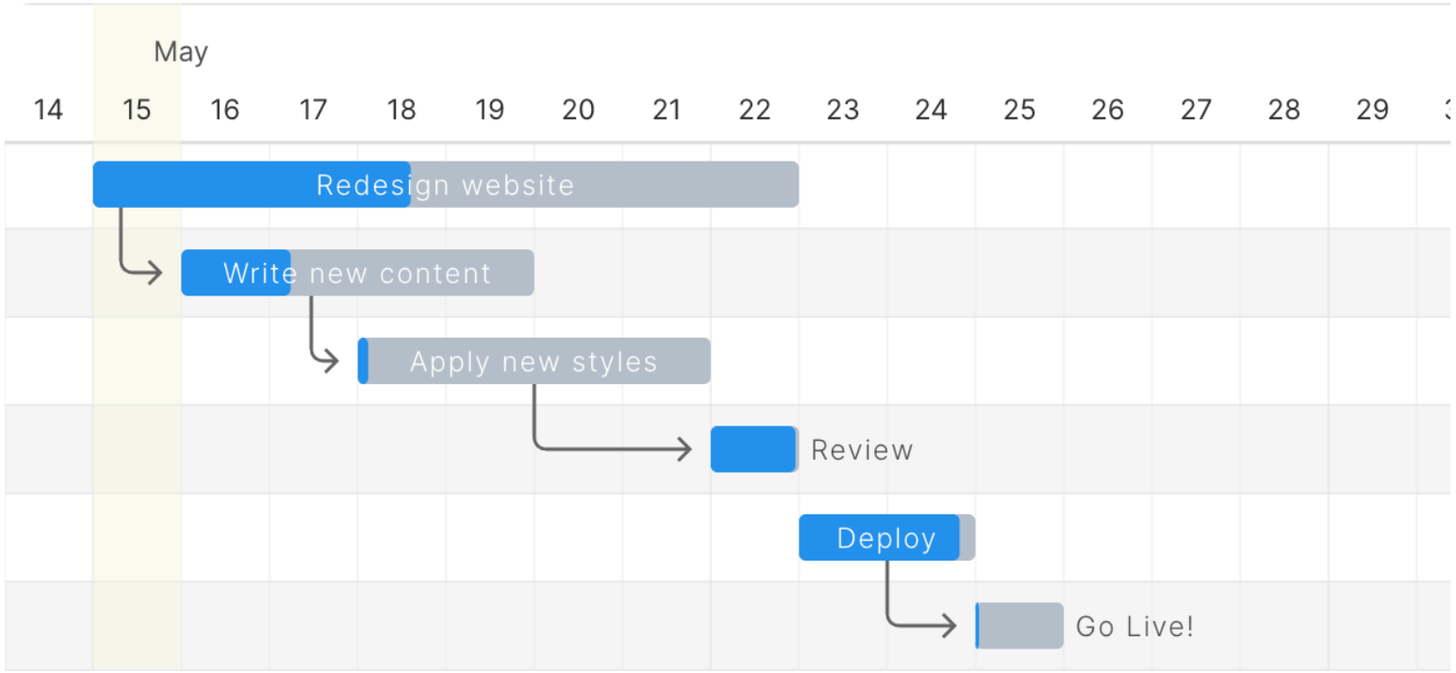
www.agametis.de

# Wer bin ich?

- Selbständiger FileMaker Entwickler im Raum München
- Beratung und Entwicklung seit über 10 Jahren
- Entwicklung von kundenspezifischen Datenbanken sowie Betreuung und Weiterentwicklung bestehender Lösungen
- FileMaker zertifiziert
- Zahlreiche Vorträge auf der FMK und dotfmp
- Web- und App-Entwicklung
- Mehr zu meinen Projekten mit Arbeitsbeispielen auf [www.agametis.de](http://www.agametis.de)

# Inhalt

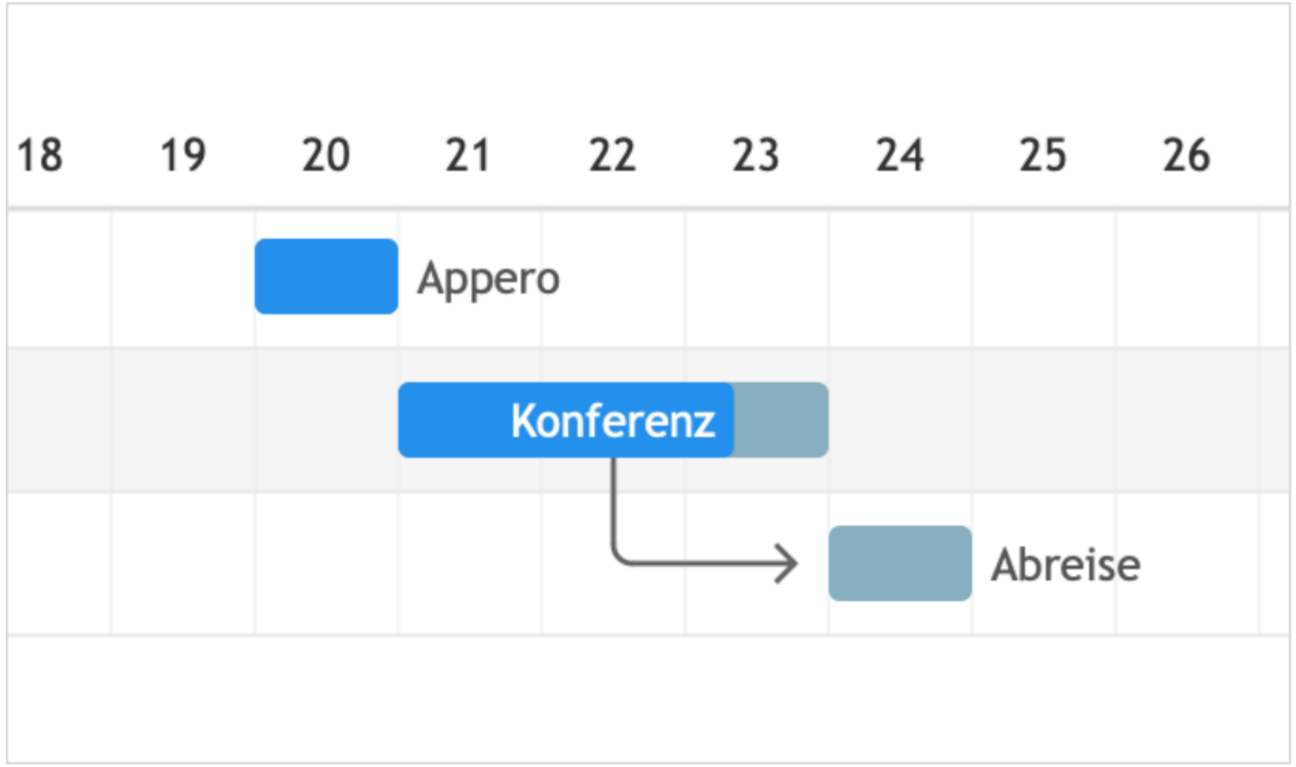
- Tooling und Debugging während der Entwicklung (damit es einfacher wird)
- Datenhandling zwischen FileMaker und JavaScript
- Einbinden einer JavaScript Bibliothek
- “frappe-gantt” als Gantt-Chart Bibliothek
- Demos
- FAQ



# Wo wollen wir hin?

| id | name | start | end | progress | dependencies | project |
|---|---|---|---|---|---|---|
| Task 1 | Appero | 20.06.2023 | 20.06.2023 | 100 | | FMK 2023 |
| Task 3 | Konferenz | 21.06.2023 | 23.06.2023 | 78 | | FMK 2023 |
| Task 4 | Abreise | 24.06.2023 | 24.06.2023 | 0 | Task 3 | FMK 2023 |

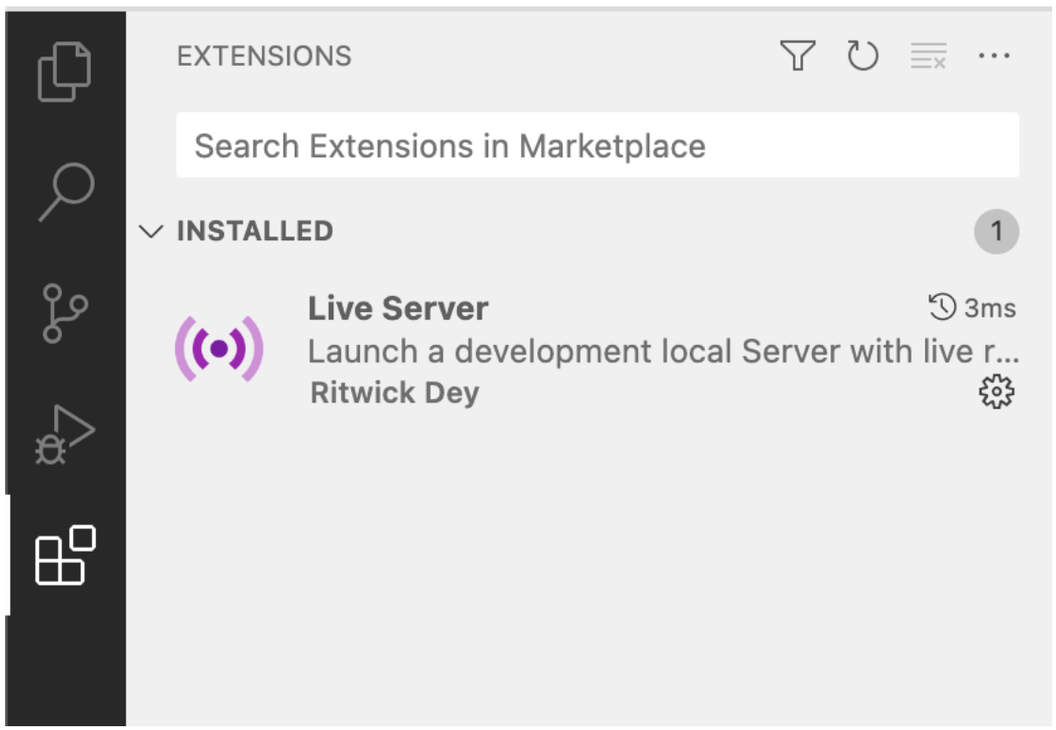
- Visualisierung von FileMaker Daten in einem Gantt-Chart
- Dynamischer/Interaktiver Datenaustausch zwischen dem Web Viewer/JS und FileMaker



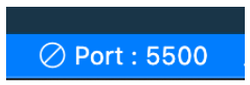
# Tooling und Debugging (während der Entwicklung)

# Tooling (der Werkzeugkasten)

- Editor: z.B. Microsoft Visual Studio Code (VSCode) https://code.visualstudio.com/
- Mit Erweiterungen können Funktionen nachinstalliert werden.
  - Webserver in Form der VSCode-Erweiterung “Live Server”




Open Source und kostenfrei

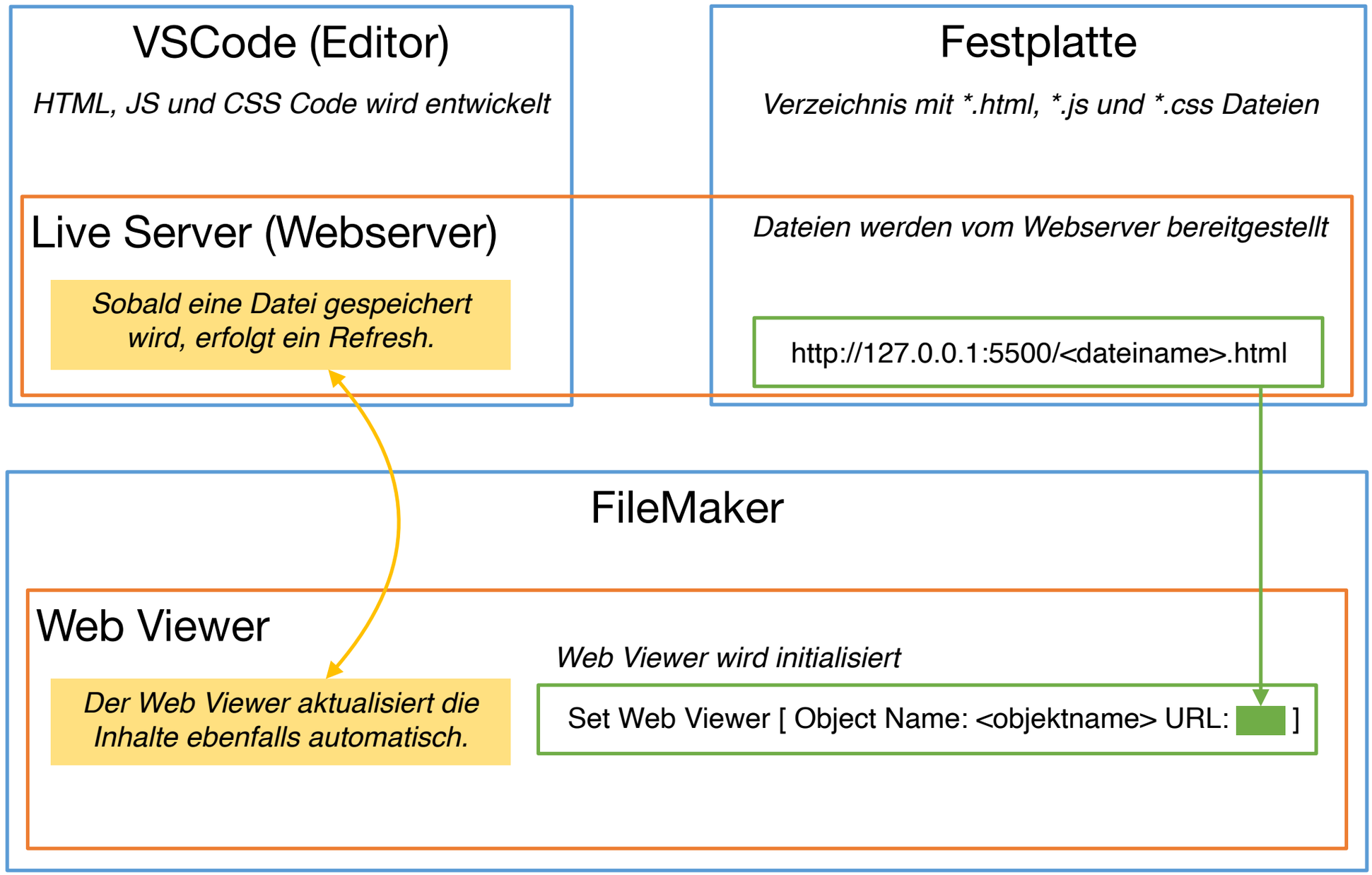

*Webserver direkt in der Statusbar starten und stoppen*

# Tooling (auf der FileMaker Seite)

- FileMaker ab 19.3 (weil, ab dieser Version quasi kein Unterschied in JavaScript zwischen Windows und Mac/iOS vorhanden)
- Der Web Viewer muss mit einem Objektnamen versehen werden, damit man ihn mit dem Skriptschritt “Set Web Viewer/Web Viewer festlegen” ansprechen kann.
  - Im Web Viewer muss in der Regel keine URL oder ähnliches angegeben werden.
  - Der Web Viewer ist quasi leer und wird zur Laufzeit initialisiert (“gefüllt”).
- Daten für den Web Viewer werden nach Möglichkeit immer mit dem Skriptschritt “Execute FileMaker Data API” geholt:
  - Grund: Daten sind sofort in einer JSON-Struktur verfügbar.
- Daten werden nach Möglichkeit immer mit dem Skriptschritt “Perform JavaScript in Web Viewer” an den Web Viewer übergeben:
  - Grund: Inhalte des Web Viewers müssen nicht vollständig geladen werden, weil nur die Daten aktualisiert werden.

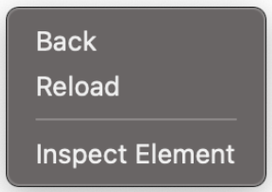
*Gilt grundsätzlich und nicht nur während der Entwicklung*

# Tooling (während der Entwicklung)

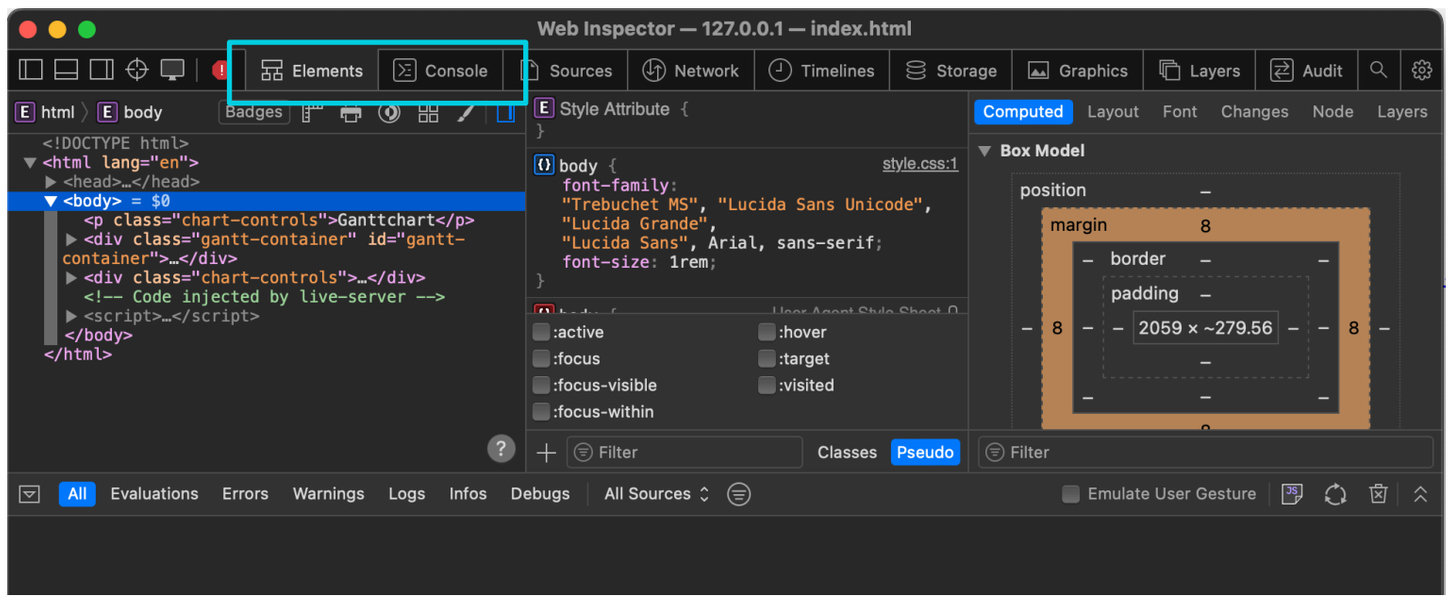


# Debugging - 1

- Debugging des JavaScript Codes direkt im FileMaker Web Viewer möglich:
  - Mit der rechten Maustaste kann der Web Inspektor (Developer Tools) aufgerufen werden (Inspect Element/Untersuchen in FM).
  - Auf dem Mac muss der Inspektor einmalig aktiviert werden. Im Terminal mit folgendem Befehl:
  - “defaults write com.filemaker.client.pro12 WebKitDebugDeveloperExtrasEnabled -bool YES”
- Unter Windows ist der Web Inspektor immer verfügbar.




*Rechte Maustaste*




*Konsole unter MacOS*

# Debugging - 2

Mit “console.log(daten)” (in der Konsole) oder “alert(daten)” (als Dialog) können Informationen (z.B. Zwischenstände) im Web Viewer mit JavaScript ausgeben werden.

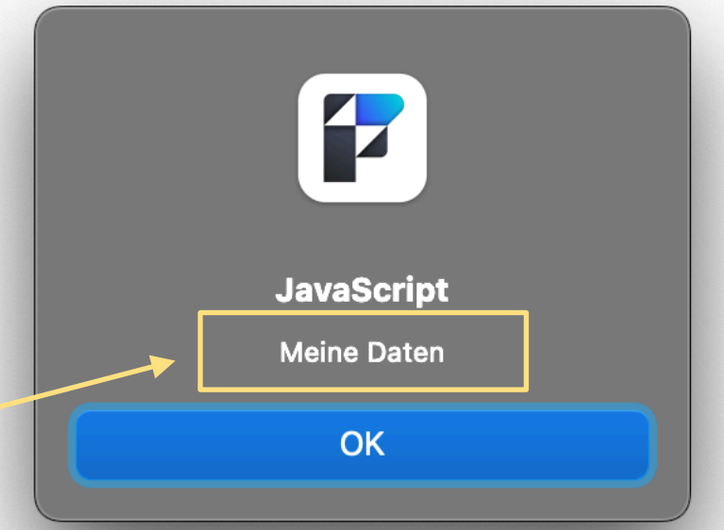
Ausgabe in der Konsole

Meine Daten

```
131
132   console.log("Meine Daten");
133
134   alert("Meine Daten");
135
```

*JavaScript Code*

Ausgabe im Dialog




Inhalt ist markierbar und kopierbar!

# Datenhandling und Einbinden von JavaScript-Bibliotheken

# JSON und JavaScript-Objekte

- JSON = "JavaScript Object Notation”

  => muss also etwas mit einem Objekt in JavaScript zu tun haben

JSON-Struktur

```
{
“foo”: “bar”
}
```

- bei der Notation eines JS-Objekts wird der "Schlüssel" normalerweise ohne Anführungsstriche geschrieben (sonst ist alles quasi gleich wie bei JSON)

JS-Objekt

```
{
foo: “bar”
}
```

# Parameterübergabe zwischen FM und JS (Fortgeschritten)

- In FileMaker kann ein Parameter mit allen Daten als JSON-Text an die JavaScript-Funktion übergeben werden.
- In JavaScript wird der JSON-Text dann in ein JavaScript-Objekt umgewandelt.

    - In FileMaker $parameter als JSON-Text

        => Im Web Viewer/JS wird er mit dem Befehl JSON.parse(parameter) in ein JS-Objekt umgewandelt

- Umgekehrt, werden alle Daten in JavaScript, die an FileMaker übergeben werden, wiederum als JSON-Text zurückübergeben

    - Im Web Viewer/JS wird das JS-Objekt mit dem Befehl JSON.stringify(parameter) umgewandelt

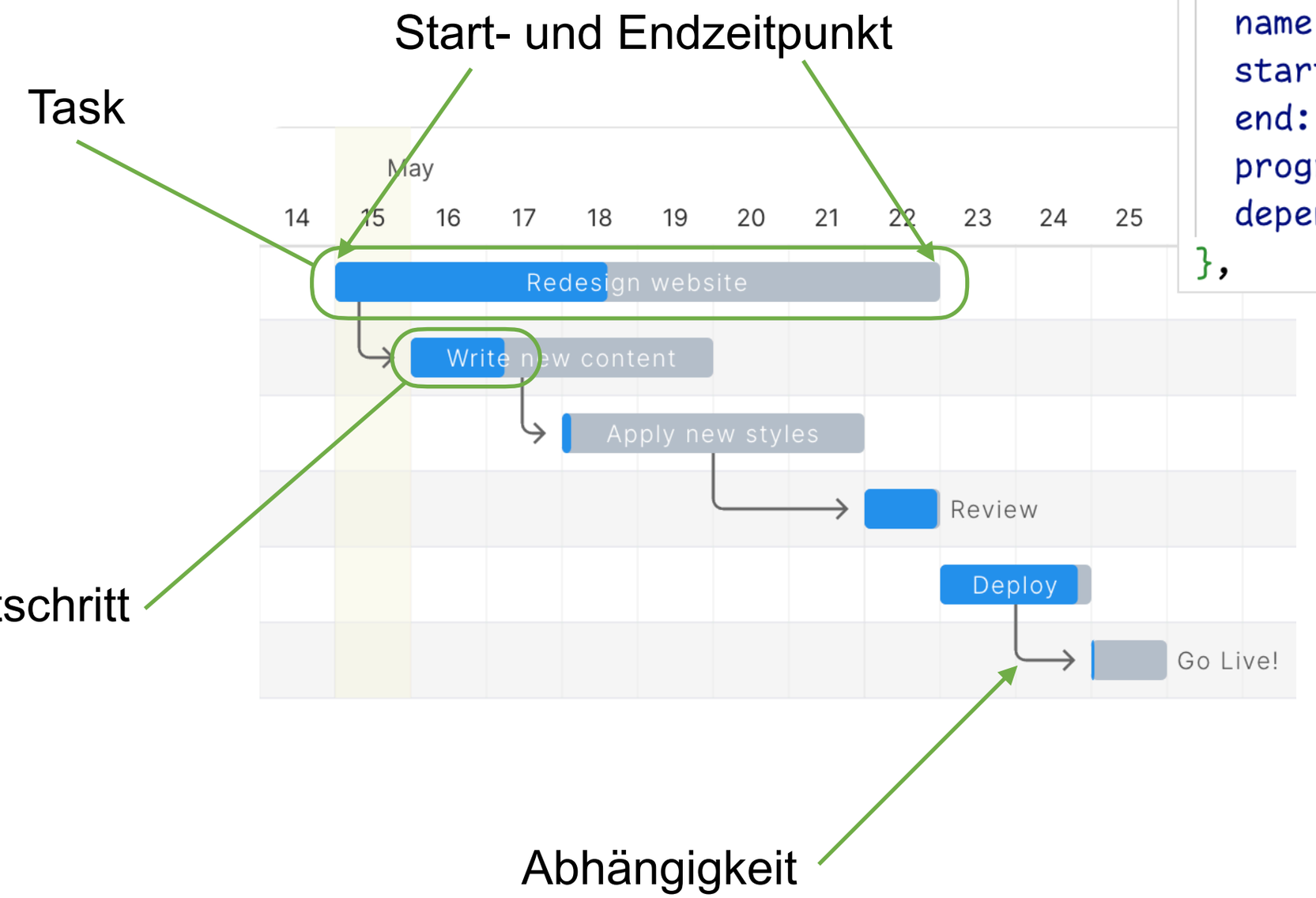
        => In FileMaker kommt der $parameter als JSON-Text an

# Einbinden einer JS-Bibliothek

- als lokale Datei mit Bezug auf die Festplatte (in der Regel relativ zur html-Datei)
  - Pro:
    - Dateien sind lokal
    - unabhängig von anderen Servern
    - unter eigener Kontrolle
- als Verweis auf ein Repository (CDN=Content Delivery Network)
  - https://cdnjs.com/ (Beispiel für ein CDN)
  - https://cdnjs.com/libraries/frappe-gantt (Übersicht)
  - Kontra:
    - Dateien könnten kompromittiert sein,
    - nicht unabhängig
    - nicht unter eigener Kontrolle

# frappe-gantt Bibliothek

# Gantt-Chart Eigenschaften




```
{
  id: "Task 1",
  name: "FMK2023",
  start: "2023-06-21",
  end: "2023-06-24",
  progress: 80,
  dependencies: "Task 2, Task 3",
},
```

*Ein Task als JavaScript-Objekt*

# Gantt-Chart Bibliothek - 1

- Gantt-Chart Bibliothek von Frappe https://frappe.io/gantt
- einfache Bibliothek für einfache Gantt-Diagramme
  - “A simple, interactive, modern gantt chart library for the web with drag, resize, dependencies and time scales”
- Drei Dateien werden mindestens benötigt: JS für die Funktionalität und CSS für die Optik
  - Links zu den Dateien in einem CDN:
    - https://cdnjs.cloudflare.com/ajax/libs/frappe-gantt/0.6.1/frappe-gantt.min.js
    - https://cdnjs.cloudflare.com/ajax/libs/frappe-gantt/0.6.1/frappe-gantt.min.css
    - https://cdnjs.cloudflare.com/ajax/libs/snap.svg/0.5.1/snap.svg-min.js

# Gantt-Chart Bibliothek - 2

- Wie nähert man sich der Bibliothek?
  - Homepage des Projektes: https://frappe.io/gantt
- Dokumentation mit “Getting Started”oder Beispielen helfen beim Einstieg.
- JS- und CSS-Basiskenntnisse sind sehr von Vorteil (aber nicht unbedingt notwendig um anzufangen).
- “Der Appetit kommt beim Essen” und so auch hier. Je tiefer man drin ist, desto mehr möchte man umsetzen.

# Interaktion mit der Gantt-Chart Bibliothek

- Neuer Task wird mit einem FileMaker-Knopf und mit Hilfe eines FileMaker-Layouts im Kartenfenster (card window) erzeugt (separates Fenster bietet sich an, da man so das Layout mit dem Web Viewer nicht verlassen muss).
- Bearbeiten eines Tasks erfolgt aus dem Web Viewer heraus:
  - Daten werden in FileMaker in einem Kartenfenster bearbeitet.
  - Nach den Änderungen/Update der Daten kann man entweder:
    - alle Tasks im Web Viewer aktualisieren oder
    - nur den veränderten/neuen Task aktualisieren - wenn bei großen Datenmengen (vielen Tasks) das Aktualisieren eventuell länger dauern sollte.
- Grundsätzlich: die Daten nach Möglichkeit immer mit dem Skriptschritt “Perform JavaScript in Web Viewer” an den Web Viewer übergeben, damit der Web Viewer nicht jedes Mal vollständig geladen/initialisiert werden muss.

# Herausforderungen beim Datumsformat

- Beim Datenaustausch zwischen unterschiedlichen Systemen muss man in der Regel auf die Formatierung des Datums/Zeitstempels achten.
- FileMaker arbeitet für gewöhnlich mit dem US Format (MM/TT/JJJJ).
- Seit FileMaker 2023 (FileMaker Pro 20.x) haben wir nun beim Skriptschritt “Execute FileMaker Data API” die Möglichkeit die Formatierung zu beeinflussen.
  - Neuer Parameter “dateformats” mit den drei Optionen
  - “0” - US Format (also wie bisher => default Ausgabe wenn nicht gesetzt)
  - “1” - Datumsformat der Datei/Feldes (beim Initialisieren der Datei automatisch festgelegt)
  - “2” - ISO8601 (JJJJ-MM-TT => ein Segen)
  - Vorsicht: der Parameter dateformats erwartet die übergebene Zahl als String (mit “JSONString” als Formatangabe ist es immer gegeben)

# Demo

# Die Demos

- Demo1: Einbindung der Bibliothek nach Anleitung auf der Homepage des Projektes.
- Demo2: Erweiterung um einfache Integration mit Daten aus FileMaker.
- Demo3: Vollständige Integration in FileMaker:
  - Lokale JS-Bibliotheken werden bei Bedarf aus Container-Feldern geladen.
  - In FileMaker gespeicherte Daten werden im Gantt-Diagramm dargestellt.
  - Interaktive Erstellung und Modifikation von Daten.

# Downloads

https://ag.amet.is/fmk2023

# FAQ

Vielen Dank für euer Interesse!

# Vielen Dank unseren Sponsoren

MonkeyBread Software

Claris

FileMaker Magazin

DATATRUST
BACKUP & ARCHIVIERUNG

24U SOFTWARE

gfm gofilemaker.de
FileMaker Software & News

Ottomatic
MODERN
FILEMAKER HOSTING

JK SOLUTION

karo.at
FileMaker
Development