

LB02 - Leistungsbeurteilung Vue.js Projekt coden (ME23d)



Hinweis: Die Leistungsbeurteilung findet Ende Juni statt. Dieses Dokument gibt Ihnen bereits jetzt einen vollständigen Überblick, damit Sie sich optimal vorbereiten können.

Was Sie erwartet

Sie erhalten zu Beginn der Leistungsbeurteilung:

- Ein **Starter-Projekt** (Vue.js bereits konfiguriert, npm install noch nötig)
- Ein **Figma-Design** mit dem zu bauenden Interface
- **Daten** als Array, die Sie anzeigen sollen
- Ein **kurzes Anleitungsbblatt** mit der groben Vorgehensweise

Die Art der Aufgabe ist Ihnen bekannt – sie ist ähnlich aufgebaut wie die Live-Coding-Anleitungen aus dem Unterricht. Der Inhalt (Design, Daten) ist neu, das Muster kennen Sie bereits.

Was Sie umsetzen müssen

Anforderung	Beschreibung
Komponenten	Mindestens 2 eigene .vue-Dateien (ohne App.vue) erstellen
Array	Daten in einem Array speichern
Anzeige	Daten mit v-for als Liste oder Karten (=rechteckige Divs) darstellen
Props	Daten von einer Eltern- an eine Kind-Komponente übergeben
Reaktivität	ref() und Bindings ({{ }}, :, v-bind) korrekt einsetzen
Event	Mindestens ein @click oder v-on-Event mit einer Funktion
Styling	Komponenten mit Flexbox oder CSS-Grid layouten und nach dem Design gestalten (Fonts, Farben) (keine Responsiveness - Mobile only)
Transition	einen Übergang zwischen zwei Zuständen mit CSS-Transition animieren

Zeitplan

Zeit	Phase
10 Min.	Starterpaket mit Code und Figma downloaden, Code starten mit npm install & npm run dev, Anleitungsblatt studieren, Fragen stellen – kein Coding
60 Min.	Coding
5 Min.	Abgabe vorbereiten und in Moodle hochladen



Wichtig: 60 Minuten effektive Coding-Zeit sind knapp. Schauen Sie regelmässig auf die Uhr. Wenn etwas zu lange nicht funktioniert: kommentieren Sie es aus und arbeiten Sie weiter. Nicht hängen bleiben.

Erlaubt	Nicht erlaubt
Google (nur mit Web) aktiviert / DuckDuckGo (keine KI-Funktionen → Erklärung unter dieser Tabelle)	Bing, Google Suche mit „KI Übersicht“, ChatGPT, Claude, Gemini, Copilot, Junie oder andere KI
MDN Web Docs (mdn.dev)	Code von anderen Lernenden oder externen Personen (live-Kopien, Duplikate)
Vue.js Dokumentation (vuejs.org)	Figma-to-Code oder Design-To-Code Plugins
Moodle-Inhalte (moodle.bzz.ch) und eigene Notizen	KI-Extensions im Editor
Code-Snippets (= Autocomplete im Editor ≤ 7 Zeilen)	Chats-Fenster offen lassen (Teams, WhatsApp, Chatbots etc.)

Websuch-Maschinen

Nicht erlaubt:

Google Suche mit KI-Übersicht aktiv

Google

const data = '' weatherData.value = data /* 2. Ersetzen Sie den leeren String (und die '') dur

Alle Bilder Shopping Videos Kurze Videos News Mehr Suchfilter

Diese Ansicht ist nicht erlaubt

Übersicht mit KI

Ersetzen Sie den leeren String durch das Ergebnis von `await response.json()`. Da `.json()` asynchron ist, muss auch die `await`-Anweisung vor dem `response.json()` verwendet werden, um das Auslesen abzuwarten.

Der korrigierte Code sieht wie folgt aus:

```
javascript
const data = await response.json();
weatherData.value = data;
```

Spezifisch generierter Code auf die Suchanfrage: darum nicht erlaubt

Mehr anzeigen

How do I fetch JSON using JavaScript Fetch API? - ReqBin

24.11.2023 — To fetch JSON data using JavaScript's `async` and `await` operators, you must first call `await fetch()` to resolve it to a `Response`...

ReqBin

Fetch: POST JSON data - javascript - Stack Overflow

("passageText": "JSON POST with Fetch: To send JSON data using the 'fetch' API stringify the JSON object using 'JSON.stringify()' ...

stackoverflow.com

Bing Suche (Microsoft)

Bing

const data = '' weatherData.value = data /* 2. Ersetzen Sie d

ALL SEARCH IMAGES VIDEOS MAPS NEWS COPILOT MORE

Bing (Suche von Microsoft): nicht erlaubt, weil sich KI-Funktion nicht ausblenden lässt.

Du möchtest also den leeren String '' durch die tatsächlich ausgelesenen JSON-Daten aus einer HTTP-Response ersetzen. Das bedeutet, du musst **asynchron** auf die Antwort warten und dann den Body mit `.json()` parsen. Hier ein vollständiges Beispiel in JavaScript (z. B. für Vue oder reines JS):

```
JavaScript
// Beispiel: Wetterdaten laden und in weatherData.value speichern
async function loadWeatherData() {
  try {
    // 1. Anfrage an API senden
    const response = await fetch('https://api.example.com/weather');

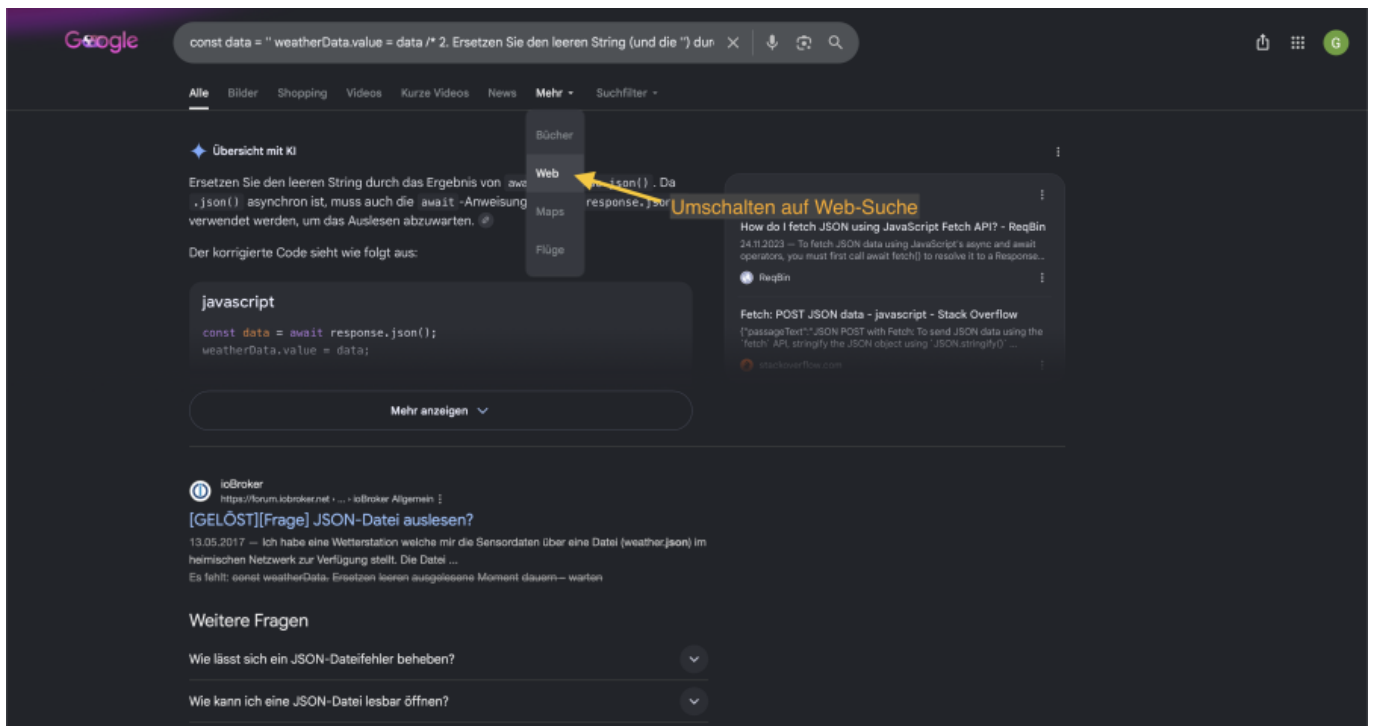
    // 2. Prüfen, ob die Antwort OK ist
```

Copy code

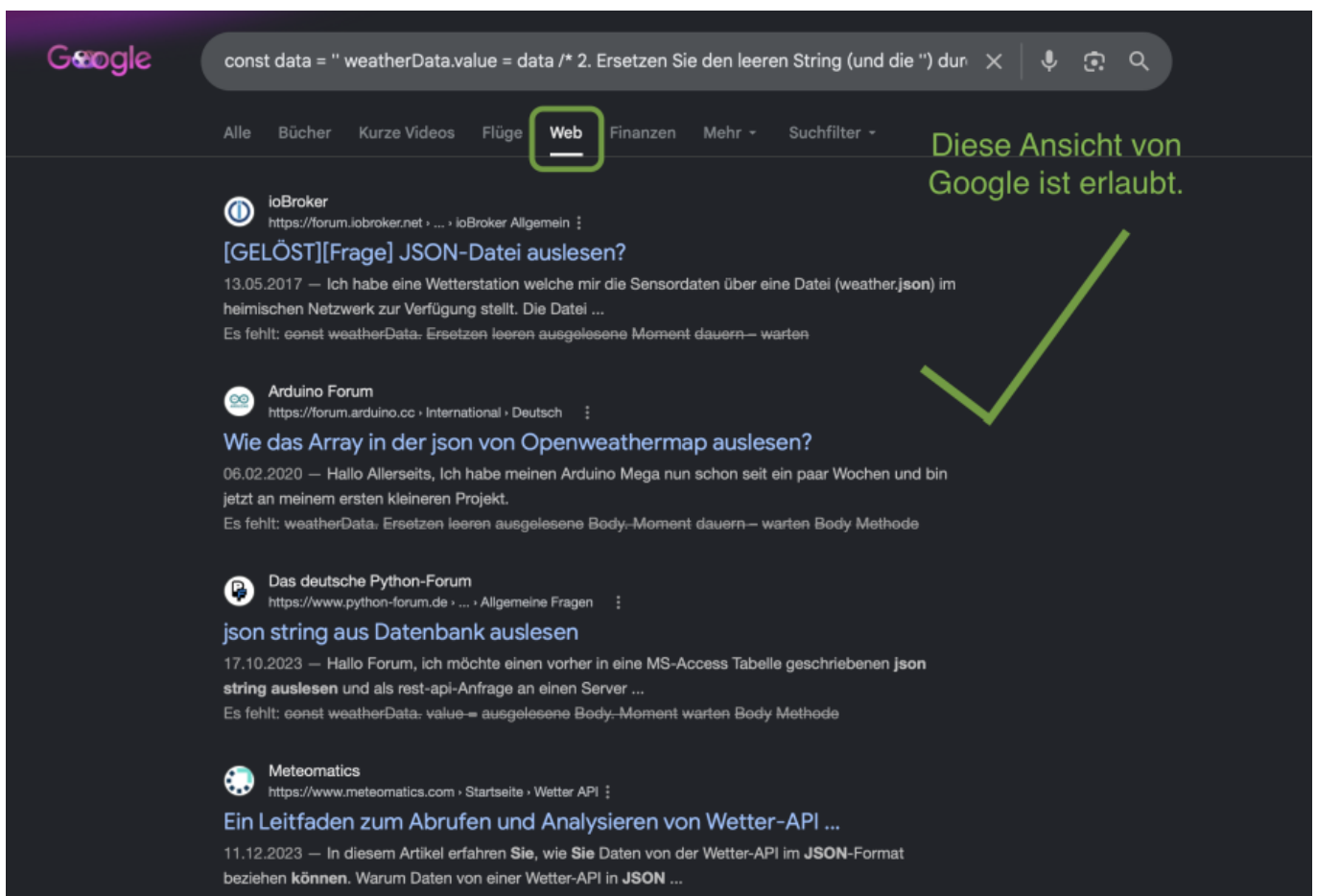
What would you like to adjust? Rewrite Testing Tools More Actions Try more templates

Erlaubt:

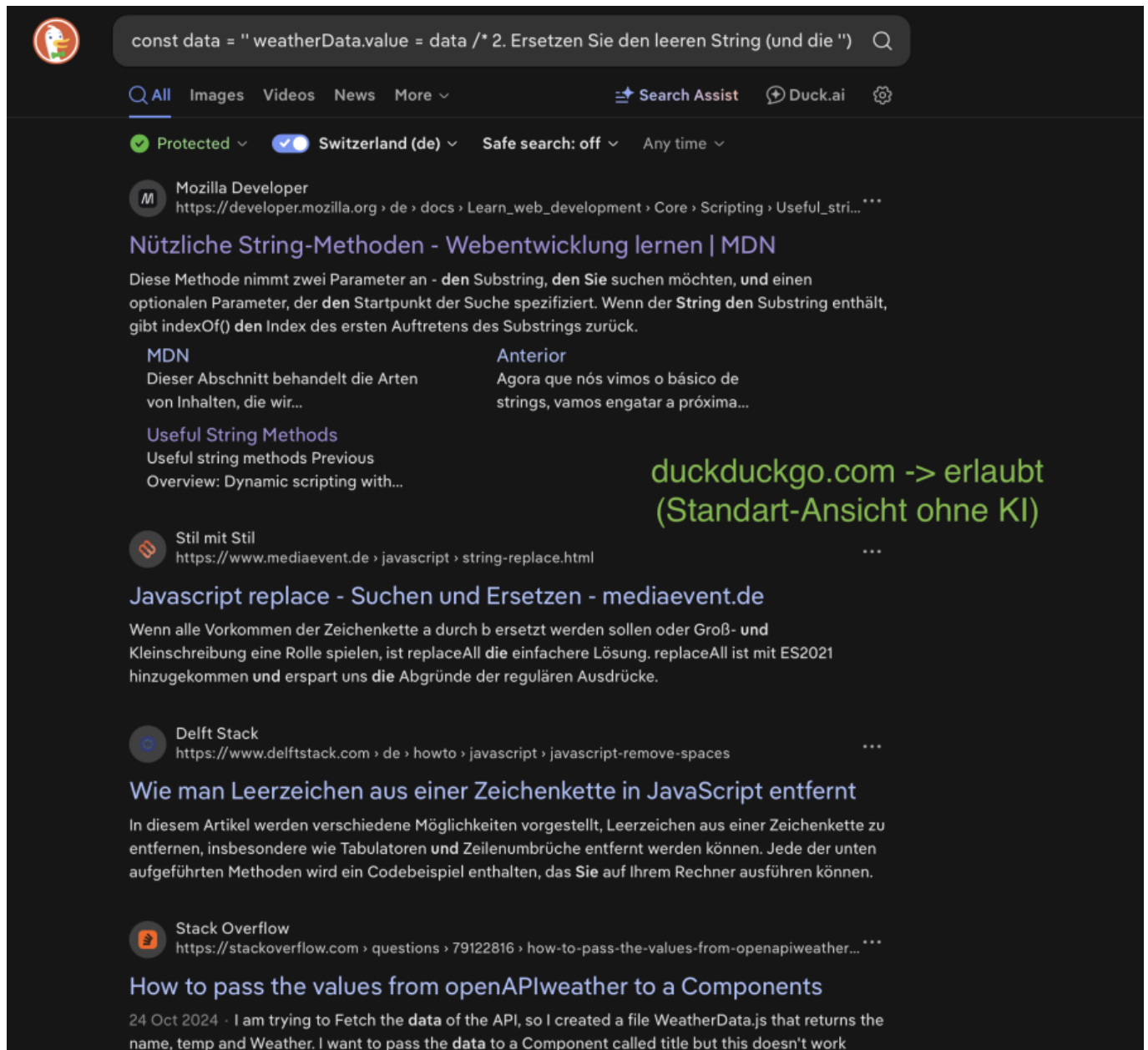
Google Suche: KI-Übersicht abschalten



Google Suche: ohne KI Übersicht



Duckduckgo: ohne KI Übersicht



The screenshot shows a DuckDuckGo search interface with a dark theme. The search bar at the top contains the text: `const data = " weatherData.value = data /* 2. Ersetzen Sie den leeren String (und die '')`. Below the search bar, there are filters for "Protected", "Switzerland (de)", "Safe search: off", and "Any time". The search results are displayed in a list format. The first result is from Mozilla Developer, titled "Nützliche String-Methoden - Webentwicklung lernen | MDN". The second result is from mediaevent.de, titled "Javascript replace - Suchen und Ersetzen - mediaevent.de". The third result is from Delft Stack, titled "Wie man Leerzeichen aus einer Zeichenkette in JavaScript entfernt". The fourth result is from Stack Overflow, titled "How to pass the values from openAPIweather to a Components". A green text overlay on the right side of the screenshot reads: "duckduckgo.com -> erlaubt (Standart-Ansicht ohne KI)".

Zusätzliche Hinweise:

- Fehlermeldungen im Editor und im Browser lesen und verstehen (→ LU07)
- Prettier (VS Code) oder Auto-Format (Webstorm) aktiv lassen – sauberer Code macht Bugs sichtbar
- Computer **vollständig geladen** (Akku) mitbringen
- **Bildschirm hell einstellen**, damit die Lehrperson sehen kann, was Sie tun
- Keine Screen-Filter (Polarfilter o.ä.), die den Blick von der Seite blockieren → falls nicht entfernbar: Schulcomputer im Voraus einrichten und üben

Konsequenzen bei Betrug

Wird der Einsatz von AI oder das Kopieren von Code erkannt oder begründet vermutet: In diesem Fall erfolgt eine mündliche Nachprüfung (LB04), die 50% der Modulnote ausmacht.

Abgabe

Ihr Projekt als .zip-Datei in Moodle hochladen:

`vorname_nachname_klassenname_LB02_M291.zip`

Wichtige Regeln:

- `node_modules`-Ordner **nicht** einschliessen → Nicht umgesetzt, kostet Punkte
- `.idea`-Ordner **nicht** einschliessen → Nicht umgesetzt, kostet Punkte
- Keine Umlaute oder Sonderzeichen im Dateinamen: ä → ae, ö → oe, ü → ue, é → e usw. → Nicht umgesetzt, kostet Punkte
- **Nur** den Vue.js-Projektordner zippen – nichts anderes → Nicht umgesetzt, kostet Punkte

Vorbereitung - was Sie jetzt tun sollten

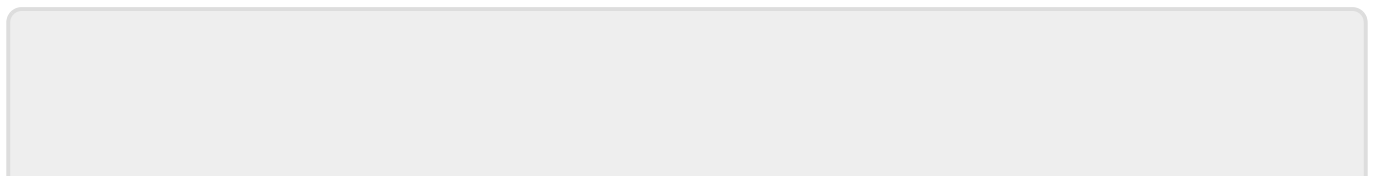
Nutzen Sie die verbleibende Zeit, um sich gezielt vorzubereiten. Die folgenden Punkte sind prüfungsrelevant:

- `npm install` und `npm run dev` im Terminal im Projektordner ausführen können
- Das Vue-Projekt im Browser anzeigen (via `localhost:5173`)
- Neue Komponenten (`.vue`-Files) erstellen und importieren können
- Grundstruktur von Komponenten erzeugen (`<script setup>`, `<template>`, `<style>`)
- Daten mit `v-for` und Props darstellen können
- ZIP-Datei korrekt erstellen (ohne `node_modules`) und in Moodle hochladen üben
- Inhalte zu Vue.js, Frameworks, CSS repetieren

Fragen

Zu Beginn der Prüfung haben Sie **10-15 Minuten** Zeit, Figma-Design und Anleitungsblatt zu lesen und Fragen zu stellen. Diese werden im Plenum beantwortet. Nach dem Start der Prüfung ist keine individuelle Unterstützung durch die Lehrperson mehr möglich – üben Sie deshalb das selbstständige Debuggen.

Bei Fragen zum Briefing wenden Sie sich jetzt an die Lehrperson – nicht erst kurz vor der Leistungsbeurteilung.



From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

https://wiki.bzz.ch/de/modul/m291/leistungsbeurteilungen/02_lb/a_briefing_me23d?rev=1781809480

Last update: **2026/06/18 21:04**

