**ASP.NET Core WebAPI**

**Grundlagen: Was ist eine WebAPI?**

* Eine **WebAPI** ist eine Schnittstelle, über die Programme Daten austauschen.
* In ASP.NET Core wird die WebAPI meist über **HTTP** umgesetzt.
* Eine Ressource (z. B. ContactInfo) wird über eine URL identifiziert.

Beispiel:

* Ressource: ContactInfo
* URL: https://localhost:5001/api/contactinfos

**REST-Prinzipien**

**Ressourcenorientierung**

* Jede Entität (z. B. ContactInfo) wird als **Ressource** behandelt.
* Jede Ressource hat eine eindeutige URL:
  + /api/contactinfos (alle)
  + /api/contactinfos/5 (ein bestimmtes Objekt)

**HTTP-Verben als Aktionen**

* **GET** → Daten lesen
* **POST** → neue Ressource erstellen
* **PUT** → Ressource aktualisieren
* **DELETE** → Ressource löschen

**REST vs. RESTful**

* **REST**: Architekturprinzipien (z. B. Ressourcen, HTTP, Statuscodes).
* **RESTful**: strikte Anwendung dieser Prinzipien.
  + Richtige Verwendung von URLs, Verben und Statuscodes.
  + Beispiel: DELETE /api/contactinfos/5 gibt 204 No Content zurück, nicht einfach nur 200 OK.

**Statuscodes (wichtig in RESTful APIs)**

* **200 OK** → Erfolg (bei GET oder nach Update).
* **201 Created** → neue Ressource wurde erfolgreich erstellt.
* **204 No Content** → Ressource erfolgreich gelöscht oder aktualisiert, keine Daten im Body.
* **400 Bad Request** → ungültige Anfrage.
* **404 Not Found** → Ressource nicht vorhanden.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Swagger (OpenAPI)**

**Was ist Swagger?**

* Swagger ist ein **Werkzeug zur Dokumentation** von WebAPIs.
* In ASP.NET Core kommt es über das Paket **Swashbuckle.AspNetCore**.
* Es erzeugt eine **automatische Benutzeroberfläche** (UI), in der alle Endpunkte aufgelistet sind.

**Vorteile von Swagger**

* **Selbstbeschreibung**: Die API erklärt sich selbst.
* **Interaktiv**: Man kann Endpunkte direkt im Browser testen (z. B. POST mit JSON-Body).
* **Standardisiert**: Export im **OpenAPI-Format**, wodurch andere Tools (z. B. Postman) es nutzen können.

**Aktivierung in ASP.NET Core**

In Program.cs oder Startup.cs:

builder.Services.AddSwaggerGen();

...

app.UseSwagger();

app.UseSwaggerUI();

**Ergebnis**

* Aufrufbar unter /swagger im Browser.
* Zeigt automatisch alle Controller und Methoden (GET, POST, PUT, DELETE).
* Schüler können direkt ausprobieren: z. B. POST /api/contactinfos mit JSON-Daten.