

Datenschnittstellen

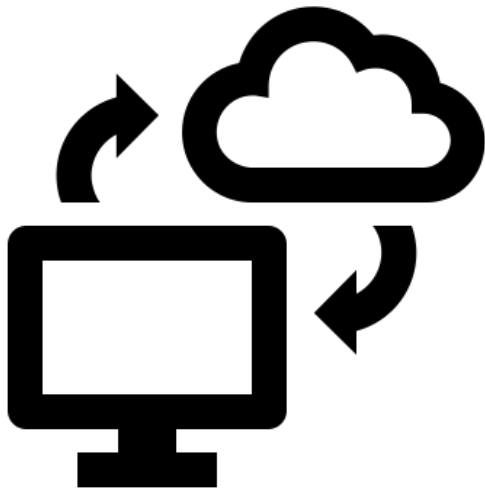
1

RESTful API

API

Application Programming Interface
= Programmierschnittstelle

- jedes Betriebssystem bietet APIs, um auf das System zuzugreifen, z.B. auf das Dateisystem, um einen Sound abzuspielen, ...
- im weiteren Sinne ist die Definition einer Python-Bibliothek, z.B. Pandas, auch eine API
- viele Online-Dienste bieten eine API. Diese nennt man auch Webservice



RESTful API (oder auch nur REST-API)

= Representational State Transfer

ist ein Software-Konzept für verteilte Systeme,
insbesondere Webservices

- Client-Server
- Zustandslosigkeit
- Einheitliche Schnittstelle
- Caching
- Mehrschichtige Systeme
- Code on Demand (optional)

Client-Server-Architektur

Der Server stellt einen Dienst bereit, der von Clients angefragt werden kann

Zustandslosigkeit

Jede Nachricht enthält alle Informationen, die für den Client oder Server notwendig sind, um die Nachricht zu verstehen.

Es müssen keine Zustände zwischen zwei Nachrichten gespeichert werden. Das ist vergleichbar mit Python-Funktionen, die sämtliche Parameter übergeben bekommen

Einheitliche Schnittstelle

- **Adressierbarkeit von Ressourcen:** jede Ressource hat eine URI (uniform resource identifier), z.B. "https://google.de/search?q=python"
- **Repräsentationen zur Veränderung von Ressourcen:** Auslieferung in verschiedenen Formaten möglich (JSON, XML)
- **Selbstbeschreibende Nachrichten:** HTTP-Verben wie get (siehe später)
- **Hypermedia as the engine of application state (HATEOAS):** Navigation ausschließlich über URLs, die vom Server bereitgestellt werden

Caching

Zwischenspeicherung von Daten, um unnötige Datenübertragungen und Serveranfragen zu vermeiden

Mehrschichtigkeit

Anwender wird nur eine Schnittstelle zur Verfügung gestellt. Dahinterliegende Ebenen sind verborgen

Code on Demand (optional)

Code wird im Bedarfsfall auf den Client zur lokalen Ausführung übertragen (z.B. Javascript)

- RESTful API ist ein allgemeines Konzept, aber wird in der Praxis eigentlich nur mittels HTTP (Hypertext Transfer Protocol) realisiert.
- HTTP stellt folgende Befehle (HTTP-Verben) zur Verfügung:
 - **GET**: Fordert Ressource vom Server an
 - **POST**: Erzeugt neue Ressource
 - **PUT**: Fügt Ressource hinzu oder ändert bestehende
 - **PATCH**: Ändert Teil einer Ressource
 - **DELETE**: Löscht eine Ressource
 - **HEAD**: Metadaten
 - **OPTIONS**: welche Methoden stehen zur Verfügung
 - **CONNECT**: Anfragen durch TCP-Tunnel leiten
 - **TRACE**: Rückgabe der Anfrage, um Änderungen zu checken

Rückgabecodes von APIs

Code	Bedeutung	Beschreibung
200	OK	Die Anfrage war erfolgreich
201	erzeugt	Eine neue Ressource wurde erzeugt
202	akzeptiert	Die Anfrage wurde erhalten, aber keine Änderung gemacht
204	kein Inhalt	Die Anfrage war erfolgreich, aber die Antwort enthält keinen Inhalt
400	falsche Anfrage	Die Anfrage war fehlerhaft
401	unauthorisiert	Der Client hat keine Berechtigung für die Anfrage
404	nicht gefunden	Die angefragte Ressource wurde nicht gefunden
415	Medientyp nicht unterstützt	Das Datenformat wird vom Server nicht unterstützt
422	Objekt kann nicht verarbeitet werden	Die Anfrage war im richtigen Format, aber enthält fehlende oder ungültige Daten
500	Interner Serverfehler	Der Server wirft einen Fehler beim Bearbeiten der Anfrage

Zugriff über Browser:

Anfrage in Adresszeile einfügen, z.B. `https://api.agify.io?name=holger`

Rückgabe als JSON: `{"name":"holger","age":50,"count":7333}`

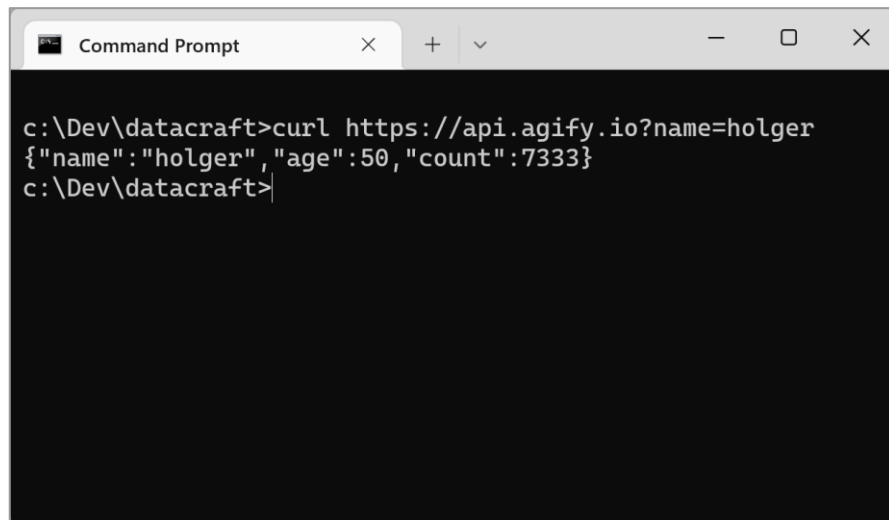
Je nach Browser weitere Darstellungen, z.B. Firefox

JSON	Rohdaten	Kopfzeilen
Speichern Kopieren Alle einklappen Alle ausklappen 🔍 JSON durchsuchen		
name: "holger"		
age: 50		
count: 7333		

JSON	Rohdaten	Kopfzeilen
Speichern Kopieren Einheitlich formatieren		
<pre>{"name": "holger", "age": 50, "count": 7333}</pre>		

JSON	Rohdaten	Kopfzeilen
Kopieren		
Response-Headers		
Access-Control-Allow-Headers	Content-Type, X-Generator-Source	
Access-Control-Allow-Methods	GET	
Access-Control-Allow-Origin	*	
Connection	keep-alive	
Content-Length	39	
Content-Type	application/json; charset=utf-8	
Date	Thu, 21 Jul 2022 08:38:46 GMT	
ETag	W/"27-D46XZm3ZELCy/DQMLjQKID0M7g"	
Server	nginx/1.16.1	
X-Rate-Limit-Limit	1000	
X-Rate-Limit-Remaining	998	
X-Rate-Reset	55273	
Request-Headers		
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	
Accept-Encoding	gzip, deflate, br	
Accept-Language	de,en-US;q=0.7,en;q=0.3	
Connection	keep-alive	
Host	api.agify.io	
Sec-Fetch-Dest	document	
Sec-Fetch-Mode	navigate	
Sec-Fetch-Site	none	
Sec-Fetch-User	?1	
Upgrade-Insecure-Requests	1	
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101 Firefox/102.0	

```
curl https://api.agify.io?name=holger
```



```
Command Prompt
c:\Dev\datacraft>curl https://api.agify.io?name=holger
{"name": "holger", "age": 50, "count": 7333}
c:\Dev\datacraft>
```

Zugriff ohne Anmeldung

Name	Beschreibung	Beispiel
7timer	Wetter	http://www.7timer.info/bin/api.pl?lon=51.551&lat=3.468&product=astro&output=json
Agify.io	Alter wird anhand des Namens geschätzt	https://api.agify.io?name=holger
Archive.org	Digitales Archiv	https://archive.org/metadata/TheAdventuresOfTomSawyer_201303
Nager	Feiertage	https://date.nager.at/api/v2/publicholidays/2022/DE
Musicbrainz	Musik	https://musicbrainz.org/ws/2/artist/5b11f4ce-a62d-471e-81fc-a69a8278c7da?fmt=json
Open Library	ältere engl. Bücher	https://openlibrary.org/api/volumes/brief/isbn/9780525440987.json
Reddit	Posts	https://www.reddit.com/r/datascience/top.json?limit=10&t=day
Wikipedia	Zugriffe	https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents/Tiger_King/daily/20210901/20210930

Einige APIs benötigen eine Anmeldung. Meist identifiziert man sich mit einem API-Key. Bei den Größeren gibt es meistens schon ein Python-Package, das den Zugriff erleichtert



- OpenWeatherMap
- API-Football
- RET Countries
- Börse / Krypto
- <https://rapidapi.com/collection/list-of-free-apis>
- <https://www.computersciencezone.org/50-most-useful-apis-for-developers/>
- <https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/>

Für Python gibt es das Package **requests**

(Installation mit `conda install requests` / `pip install requests`)

```
import requests
```

```
response = requests.get("https://api.agify.io?name=holger")
```

```
print(response.status_code) # der Status-Code
```

```
print(response.json())      # Antwort als JSON-Objekt
```

```
print(response.json()["age"]) # Zugriff auf Elemente
```

```
print(response.headers)     # Kopf der Antwort mit Informationen
```

```
print(response.content)     # rohe Bytes, z.B. bei Grafiken
```

```
print(response.text)        # Antwort als String
```