

Verwendung und Einbindung von Open-Data-Schnittstellen am Beispiel der Bundes-APIs

Tim-Jonathan Huyeng¹

¹Technische Universität Darmstadt, Institut für Numerische Methoden und Informatik im Bauwesen
E-mail(s): huyeng@iib.tu-darmstadt.de

Abstract: Aggregation von Daten mittels API-Request ist eine verbreitete Methodik zur Abfrage von Informationen aus verschiedenen Datenquellen, welche zum Beispiel in einer Applikation dargestellt werden können. Üblicherweise werden dabei die Daten für eine Anwendung in einem speziell hierfür implementierten Backend mit einer eigenen API zur Verfügung gestellt. Mit der Verabschiedung des *Zweiten Open-Data-Gesetzes 2021* wird die Menge an Daten von Bundesbehörden und öffentlichen Einrichtungen, welche frei verfügbar sind, noch deutlich gesteigert werden. Die sogenannte *Bundesstelle für Open Data (BundDEV)* hat sich zur Aufgabe gemacht, diese Daten bzw. deren API-Spezifikationen zu veröffentlichen und einfach zugänglich zu machen. Dabei wird auf den weit verbreiteten *OpenAPI*-Standard bei der Beschreibung der APIs gesetzt. Im Rahmen des Papers wird analysiert, inwiefern diese Initiative bzw. die dokumentierten APIs für Anwendungen einfach verwendet werden können. Es wird beispielhaft der Prozess für verschiedene Ausgangslagen von APIs dargelegt und beschrieben. Als Beispielanwendungsfälle werden unter anderem der Hochwasserschutz sowie dazu passende APIs herangezogen. Die öffentliche Bereitstellung von API-Spezifikationen sowie die automatische Generierung von Client-Tools ist ein weiterer Baustein für die einfache Nutzung von öffentlichen Daten. Dabei liefert der *OpenAPI*-Standard eine einfache und maschinenlesbare Möglichkeit verschiedene APIs und deren Parameter zu beschreiben.

Keywords: Open-Data, APIs, Hochwasserschutz, Automatisierung

1 Einleitung

Daten sowie die Verfügbarkeit von Daten sind wichtige Bestandteile in den Digitalisierungsstrategien von vielen Bereichen [1, S. 34]. Dies gilt ebenfalls für die Bauindustrie, in der die Verwendung und Analyse von Datenquellen aufgrund des technologischen Fortschritts in den letzten Jahren ein zentraler Baustein geworden ist. Neben privaten und proprietären Daten hat die Verwendung öffentlich verfügbarer Informationen ein großes Potenzial. Neben den technischen Weiterentwicklungen im

Bereich der Datenerfassung gibt es zudem auch politische Motivation und entsprechende Gesetze, die insbesondere öffentliche Einrichtungen bei der Bereitstellung der Daten mehr in die Pflicht nehmen.

Damit die Daten von Anderen verwendet werden können, werden diese entweder zum Download bereitgestellt oder mittels einer Schnittstelle (API - Application Programming Interface) über das Web zur Verfügung gestellt. Die Bereitstellung mittels einer API ist vor allem sinnvoll, wenn aktuelle (live) und nicht nur historische Daten zur Verfügung gestellt werden und eine Anbindung der Daten an andere Tools oder Webservices ermöglicht werden soll. Im Rahmen des Papers werden daher nur Schnittstellen nicht aber die einfache Bereitstellung als Datei-Download beachtet. Die Dokumentation von APIs ist auf verschiedene Art und Weisen möglich und an den Anspruch sowie die Möglichkeiten des Anbieters gebunden. Ist es ein konkretes Ziel der Betreiber Daten anderen Personen zur Verfügung zu stellen, ist meist eine ausführliche Dokumentation oder Anleitung zur Verwendung der API vorhanden. Diese können entweder als reine Textdokumentation oder auch über sonstige maschinenlesbare Formate (z.B. *OpenAPI*) bereitgestellt werden. Üblicherweise werden die Routen, die Parameter sowie die zu erwartenden Antworten beschrieben. Ist es vom Betreiber nicht vorgesehen, oder hat der Betreiber kein gesteigertes Interesse an einer einfachen Nutzung seiner Schnittstelle ist es möglich, dass keine Dokumentation zur Verfügung steht. Durch Reverse Engineering und die Verwendung von verschiedenen Tools ist es allerdings dennoch möglich die API-Struktur nachzuvollziehen und sich darauf aufbauend eine Dokumentation zu erstellen. Diese kann anschließend anderen Personen öffentlich zur Verfügung gestellt werden. Im Rahmen des Papers wird für diese Möglichkeiten jeweils ein Beispiel beschrieben.

Als einheitliches Dokumentationsformat für die im Folgenden beschriebenen APIs wird das *OpenAPI*-Format verwendet. Dieses Format wird von der *OpenAPI*-Initiative (OAI) [2], welche der *Linux Foundation* untergeordnet ist, stetig weiterentwickelt und standardisiert. In diesem textbasierten Format (*JSON* oder *YAML*) werden Informationen zu den einzelnen Routen, Parametern sowie Antworten gespeichert. Überdies können zusätzliche Beschreibungen, Informationen sowie Beispiele die Verwendung der API vereinfachen.

2 Aktueller Stand von Open-Data-Schnittstellen

Wie in der Einleitung beschrieben sind in den letzten Jahren die Möglichkeiten der Verwendung von öffentlichen Daten gestiegen. Dies liegt neben den bereits beschriebenen Gründen auch an Initiativen, welche sich zur Aufgabe gemacht haben Daten bzw. die Datenschnittstellen öffentlich zu dokumentieren. Zudem wird die Nutzbarkeit sowie Verfügbarkeit von öffentlichen Daten durch verschiedene Gesetze weiter gestärkt und vereinfacht [3]. So sind zum Beispiel die Behörden des Bundes durch das *Zweite Open-Data-Gesetz* [4] dazu verpflichtet eine öffentliche Schnittstelle über Netze zum Datenabruf bereit zustellen [5, §12a, (1)]. Neben Behörden gilt das Gesetz unter anderem auch für Unternehmen mit öffentlichen Aufträgen sowie Hochschulen und Forschungseinrichtungen.

Das Gesetz konkretisiert dies wie folgend: „[...] Daten [sind] elektronisch und in nach den anerkannten Regeln der Technik offenen, maschinenlesbaren, zugänglichen, auffindbaren und interoperablen

Formaten zusammen mit den zugehörigen Metadaten bereitzustellen“ [6, §7, (2)]. Dabei muss dies nicht erfolgen, wenn für die Behörde oder zuständige Stelle ein unverhältnismäßiger Aufwand entstehen würde. Wie der Aufwand und die daraus resultierenden Veröffentlichungen von Daten bewertet wird, muss sich in den nächsten Jahren zeigen. Untersuchungen wie zum Beispiel von Schmietendorf [7] zeigten dabei 2020 noch einige Lücken und Probleme auf.

Das Gesetz ist Teil der Digitalisierungsstrategie des Bundes, zu der auch das Portal *GovData* gehört [8]. Das Portal soll einen einfachen Zugang zu Daten und Dokumenten der öffentlichen Hand bieten. Dabei ist es möglich nach Datensätzen zu suchen und sich die entsprechenden Metadaten in den jeweils verfügbaren Formaten anzeigen zu lassen. Die Daten sowie Datenschnittstellen werden dezentral von den jeweiligen Anbietern zur Verfügung gestellt. *GovData* wurde vom IT-Planungsrat ins Leben gerufen und wird somit direkt von der öffentlich Hand betreut. Neben öffentlich gestützten Angeboten zur Sammlung von Schnittstellenbeschreibungen gibt es auch andere Initiativen nicht öffentlicher Stellen. Eine Initiative ist *BundDEV* [9], welche unter anderem von Lilith Wittmann ins Leben gerufen wurde. Die Initiative *BundDEV* soll laut den Gründern vor allem zeigen, was aktuell in den Behörden nicht umgesetzt wird und wie die Verwendung von öffentlichen Daten beschleunigt werden kann [10]. Eines der ersten Beispiele war die API-Dokumentation für das Katastrophenwarnsystem NINA [11]: Sowohl die Daten als auch die Schnittstelle zum Zugriff waren zwar seit einiger Zeit verfügbar, dies war aber mangels offizieller Unterstützung und Dokumentation kaum bekannt. Als Basis der Beschreibung wird der *OpenAPI*-Standard verwendet. *BundDEV* stellt zur Beschreibung von Schnittstellen inzwischen auch ein Template auf GitHub zur Verfügung. Mithilfe des Templates und den bereits vorhandenen Beispielen sind Beteiligungen von weiteren Personen auch ohne Expertenwissen möglich. Aktuell sind über 30 APIs dokumentiert und öffentlich einsehbar [9].

3 Konzept zur Verwendung und Einbindung von Open-Data

Die Initiative *BundDEV* zeigt, wie einfach es sein kann Schnittstellen zu dokumentieren und hierdurch für die breite Masse nutzbar zu machen. Im folgenden Kapitel soll der Workflow von Schnittstellen bis zur Verwendung dieser erläutert werden um zu zeigen, wie bei verschiedenen Ausgangslagen vorgegangen werden kann. Es werden hierfür folgende drei Ausgangslagen definiert:

Tabelle 1: Übersicht über die beschriebenen Ausgangslagen

	Dokumentation	Spezifikation
A-1	✗	✗
A-2	✓	✗
A-3	✓	✓

Hierbei ist zu beachten, dass bei der Ausgangslage A-3 die Spezifikation maschinenlesbar vorliegen muss, bei A-2 kann es möglich sein, dass die Dokumentationen auch maschinell und automatisch ausgelesen werden können, wenn sie zum Beispiel als Textdokumentation vorliegen. Nachfolgend werden

die einzelnen Ausgangslagen und die jeweilige Vorgehensweise zur Erstellung einer Dokumentation erläutert.

3.1 A-1: keine Dokumentation der API vorhanden

Ist zu einer Datenschnittstelle keinerlei Dokumentation vorhanden, gibt es verschiedene Wege diese zu erstellen. Wird die API zum Beispiel bereits auf einer anderen Webseite als Datenquelle für zum Beispiel Kartendarstellungen verwendet, können die einzelnen Requests im Browser nachvollzogen werden. Hierbei bieten sich die üblichen Entwicklertools der jeweiligen Browser an. Abhängig vom verwendeten Browser gibt es eine Ansicht, welche die einzelnen Anfragen und deren Antworten darstellt. In dieser Ansicht lassen sich Abfragen zu einer API auch mithilfe von Filtern einfach ausfindig machen und analysieren. Wird *Firefox* verwendet, ist es möglich die identifizierten Abfragen zu verändern und anschließend erneut zu senden. Mit dieser Methode können schnell und einfach Abfrageparameter getestet werden. Neben dieser sehr simplen Lösung gibt es auch weitere Tools, die bei diesem Prozess unterstützen können. Der *Swagger Inspector* [12] oder auch das von goofmint [13] entwickelte Tool „*cURL to Swagger*“ ermöglichen das Erzeugen von *OpenAPI*-Spezifikationen anhand einer Abfrage. Auch können Webtools wie *Postman* o.ä. zum Testen und Analysieren von Abfragen und deren Antworten verwendet werden. Unabhängig davon, wie hier konkret vorgegangen wird, ist das Ziel eine *OpenAPI*-Spezifikation zu erhalten, welche entweder im YAML- oder im JSON-Format gespeichert werden kann.

3.2 A-2: Dokumentation vorhanden, aber keine API-Spezifikation

Diese Ausgangslage ist insbesondere bei älteren Schnittstellen öfters der Fall. Dies kann zum Beispiel daran liegen, dass zum Zeitpunkt der Implementierung der Schnittstelle der *OpenAPI*-Standard noch nicht verbreitet war. Dokumentationen können in unterschiedlichster Qualität und Form vorliegen. Sind bei einer Dokumentation zum Beispiel neben der reinen Beschreibung auch Beispiele angegeben, erleichtert dies die Konstruktion der *OpenAPI*-Spezifikation deutlich. Eine automatische Auswertung mithilfe von Webcrawlern ist nur sinnvoll, wenn die Schnittstelle sehr umfangreich ist oder sich das hierfür entwickelte Tool auch bei anderen Schnittstellen verwenden lässt. Ist dies nicht der Fall wird empfohlen die *OpenAPI*-Spezifikation händisch anhand der Dokumentation zu erstellen.

3.3 A-3: API-Spezifikation vorhanden

Liegt eine API-Spezifikation vor, kann diese direkt im nachfolgend beschriebenen Workflow verwendet werden. Mithilfe von verschiedenen Tools wie zum Beispiel einem YAML/JSON-Linter [14] kann die *OpenAPI*-Spezifikation auf mögliche Fehler untersucht werden. Dabei ist es prinzipiell unerheblich, ob die Spezifikation von dem Anbieter der Schnittstelle oder von einer anderen Person geschrieben wurde. Anschließend kann eine Dokumentation mithilfe der *Swagger UI* im Browser zur Verfügung gestellt werden. Wird die Spezifikation in einem GitHub-Projekt verwaltet, kann z.B. *GitHub-Pages* zum Hosting der Dokumentation verwendet werden. In der webbasierten Darstellung können Abfragen ebenfalls getestet werden.

Nach Bearbeitung der jeweiligen Schritte ist das Ziel somit eine API-Spezifikation. Im hier vorgestellten Konzept ist diese in Form einer *OpenAPI*-Spezifikation gefordert. Anhand dieser Spezifikation kann dann mithilfe von Generatoren Client-Quellcode generiert werden (siehe Abb. 1).

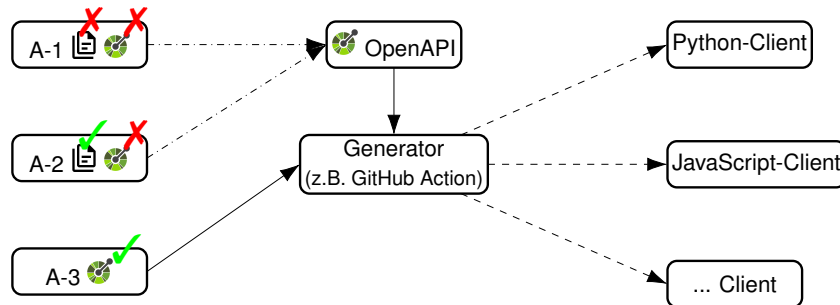


Abbildung 1: Darstellung des Workflows auf Basis der verschiedenen Ausgangslagen

4 Exemplarische Umsetzung für die verschiedenen Ausgangslagen

Die im vorherigen Kapitel beschriebene automatische Generierung des jeweils geforderten Client-Quellcodes wird in diesem Kapitel anhand einer hierfür erstellten GitHub-Action erläutert. Dabei wurde sich an der GitHub-Action, welche von *BundDEV* eingesetzt wird, orientiert [15].

4.1 A-1: Länderübergreifendes Hochwasserportal (LHP)

Als Beispiel für diesen Fall wird die API des Länderübergreifenden Hochwasserportals (LHP) verwendet. Auf der Webseite hochwasserzentralen.de werden länderübergreifende Informationen zu Pegel- und Hochwasserwarnungen zusammengefasst und auf einer Karte dargestellt. Um die Daten auf einer Karte darzustellen, werden die Informationen von einer eigenen API zur Verfügung gestellt. Hierbei werden verschiedene Datenquellen der Bundesländer und anderen öffentlichen Stellen im Backend zusammengefasst und gebündelt veröffentlicht. Im engen Sinne existiert momentan keine API, welche für die Öffentlichkeit zur Verfügung gestellt werden soll. Da die Webseite allerdings über Requests die erforderlichen Daten abfragt, kann dies auch zur Anbindung an eigene Applikationen genutzt werden. Die Dokumentation wurde hier anhand einer Request-Analyse mittels Reverse Engineering erstellt. Die zum jetzigen Zeitpunkt noch nicht vollständige Dokumentation ist auf GitHub veröffentlicht [16].

4.2 A-2: PEGELONLINE API

Die von der Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV) zur Verfügung gestellte API für Pegel in Deutschland ist in einer umfangreichen HTML-Dokumentation beschrieben [17]. Anhand dieser Dokumentation wurde eine *OpenAPI*-Spezifikation erstellt. Diese Spezifikation ist auf GitHub veröffentlicht und ist Teil des *BundDEV*-Projekts. Die API liefert umfangreiche Daten wie zum Beispiel die Wasserstände, Abflussparameter oder Wasserparameter für die überwachten Gewässer. Da diese API für den öffentlichen Zugriff entwickelt wurde bestehen aktuell keinerlei

Limitierungen oder notwendige Authentifizierungen. Abbildung 2 zeigt eine beispielhafte Anbindung über ein *Vue.js*-Frontend sowie eine Visualisierung der Daten mittels *Leaflet.js*.

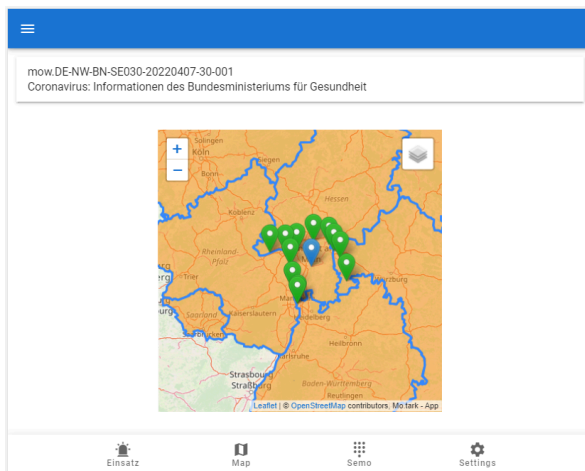


Abbildung 2: Beispielhafte Verwendung der PEGELONLINE-API in einem Frontend

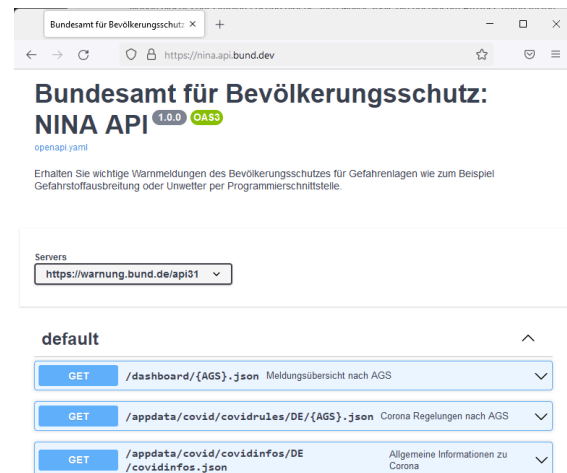


Abbildung 3: Swagger UI am Beispiel der NINA API

4.3 A-3: NINA API

Ist eine Spezifikation bereits vorhanden, kann diese direkt als Grundlage für weitere Anbindungen verwendet werden. Dabei ist es prinzipiell unerheblich, ob die Beschreibung von einer Community oder vom Anbieter der Schnittstelle zur Verfügung gestellt wird. Am Beispiel der NINA API [11] kann gezeigt werden, dass öffentliche Daten mithilfe dieser Dokumentation sehr einfach verwendet werden können. Erstaunlich ist hierbei besonders, dass für die Daten, welche zur Warnung der Bevölkerung verwendet werden, keine Dokumentation seitens des öffentlichen Anbieters existiert. Die von der Community erstellte Spezifikation ist unter nina.api.bund.dev einsehbar (siehe Abb. 3).

4.4 Automatische Generierung von Client-Code mittels GitHub-Action

Die implementierte GitHub-Action ist öffentlich verfügbar und kann dem jeweiligen Projekt einfach hinzugefügt werden. Anschließend erzeugt die Action abhängig von der Einstellung nach jedem Push oder durch manuelles Triggern aktuellen Client-Quellcode. Damit das ursprüngliche Projekt übersichtlich bleibt wird der generierte Quellcode in einen gesonderten Branch gepusht. Aktuell wird ein JavaScript- sowie ein Pythonpackage erzeugt [18].

5 Zusammenfassung und Ausblick

Die öffentliche Bereitstellung von API-Spezifikationen sowie die automatische Generierung von Client- oder Server-Tools ist ein weiterer Baustein für die vereinfachte Nutzung von öffentlichen Daten. Dabei liefert der *OpenAPI*-Standard eine einfache und maschinenlesbare Möglichkeit verschiedene APIs und deren Parameter zu beschreiben.

Im Rahmen des Papers konnte ein Workflow für die Anbindung und Verwendung von Open-Data-Schnittstellen gezeigt werden. Dieser Workflow basiert auf einer *OpenAPI*-Spezifikation der Schnittstelle und automatisch generiertem Client-Quellcode. Die Einbindung von Open-Data wird in den nächsten Jahren deutlich zunehmen, insbesondere wenn die entsprechenden Gesetze in den Behörden mehr und mehr Umsetzung finden. Dies bedeutet neben der Bereitstellung von Daten und Schnittstellen auch eine Dokumentation. Zusätzlich zu den hier gezeigten Beispiel-Schnittstellen ist davon auszugehen, dass weitere APIs mit Baubezug in den nächsten Jahren wachsen. Zum Beispiel setzt *buildingSMART* auch bei der Dokumentation des BCFs auf die *OpenAPI*-Spezifikation [19]. Da sich der Ansatz auf einheitliche Dokumentation stützt, ist dies aber auch als mögliche Schwäche zu benennen, da diese Dokumentationen teilweise erst aufwendig erstellt werden müssen. Wenn die Dokumentation zudem von der Community und nicht vom Anbieter der Schnittstelle erstellt wird, ist es möglich, dass Veränderung der API nicht direkt in der Dokumentation widerspiegelt werden.

Zudem besteht nicht bei allen Anbietern ein intrinsisches Bestreben nach einer hohen Verbreitung einer der einer Schnittstelle. Wird eine Schnittstelle umfangreich öffentlich dokumentiert, kann dies zu einer vermehrten Nutzung und damit gleichzeitig zu einem steigenden Server- und Wartungsbedarf führen.

Literatur

- [1] Bundesministerium des Innern, »Open Government Data Deutschland«, Eine Studie zu Open Government in Deutschland im Auftrag des Bundesministerium des Innern, Studie, Juli 2012.
- [2] The Linux Foundation. »OpenAPI Initiative«, OpenAPI Initiative. (2022), Adresse: <https://www.openapis.org/> (besucht am 30. 03. 2022).
- [3] BMI. »Open Data«, Bundesministerium des Innern und für Heimat. (2021), Adresse: <https://www.bmi.bund.de/DE/themen/moderne-verwaltung/open-government/open-data/open-data-node.html> (besucht am 30. 03. 2022).
- [4] *Entwurf eines Gesetzes zur Änderung des E-Government-Gesetzes und zur Einführung des Gesetzes für die Nutzung von Daten des öffentlichen Sektors*, unter Mitarb. von Bundesregierung.
- [5] *Gesetz zur Förderung der elektronischen Verwaltung (E-Government-Gesetz - EGovG)*, 25. Juli 2013. Adresse: <https://www.gesetze-im-internet.de/egovg/> (besucht am 10. 05. 2022).
- [6] *Gesetz für die Nutzung von Daten des öffentlichen Sektors (Datennutzungsgesetz - DNG)*, 16. Juli 2021. Adresse: <https://www.gesetze-im-internet.de/dng/> (besucht am 26. 05. 2022).
- [7] A. Schmietendorf, »Öffentlich angebotene Daten und Funktionen (Open Data/Open APIs) als Rückgrat des digitalen Wandels in den deutschen Behörden – eine Bestandsaufnahme«, in *Verständliche Verwaltungskommunikation in Zeiten der Digitalisierung*, R. Fisch, Hrsg., Nomos Verlagsgesellschaft mbH & Co. KG, 2020, S. 63–76, ISBN: 978-3-7489-0284-3.

- [8] IT-Planungsrat. »GovData - Datenportal für Deutschland«. (2022), Adresse: <https://www.govdata.de> (besucht am 06. 04. 2022).
- [9] bundDEV. »Entwicklungsportal - API Übersicht«, Bundesstelle für Open Data - Verwaltung Digital. (2022), Adresse: <https://bund.dev/> (besucht am 30. 03. 2022).
- [10] L. Wittmann. »Wenn die Zivilgesellschaft bei Open Data hilft«, Medium. (2021), Adresse: <https://lilithwittmann.medium.com/wenn-die-zivilgesellschaft-bei-open-data-hilft-905add0aa21b> (besucht am 28. 03. 2022).
- [11] bundDEV. »Bundesamt für Bevölkerungsschutz - NINA API«. (2021), Adresse: <https://nina.api.bund.dev/> (besucht am 29. 03. 2022).
- [12] SmartBear Software. »Swagger Inspector«. (2022), Adresse: <https://inspector.swagger.io/builder> (besucht am 12. 05. 2022).
- [13] goofmint. »cURL to Swagger«. (2022), Adresse: <https://goofmint.github.io/cURLtoSwagger/> (besucht am 01. 05. 2022).
- [14] Stoplight. »Spectral - Open Source API Description Linter | Stoplight«. (2022), Adresse: <https://stoplight.io//open-source/spectral> (besucht am 02. 05. 2022).
- [15] wirthual, *deutschland-generator-action*, 2021. Adresse: <https://github.com/wirthual/deutschland-generator-action> (besucht am 09. 05. 2022).
- [16] T.-J. Huyeng. »Hochwasserzentralen - OpenAPI Documentation«. (2022), Adresse: <https://t-huyeng.github.io/hochwasserzentralen-api/> (besucht am 25. 05. 2022).
- [17] Wasserstraßen- und Schifffahrtsverwaltung des Bundes (WSV). »PEGELONLINE REST-API Dokumentation«. (2022), Adresse: <https://www.pegelonline.wsv.de/webservice/dokuRestapi> (besucht am 12. 05. 2022).
- [18] T.-J. Huyeng. »GitHub OpenAPI Action«. (2022), Adresse: <https://github.com/t-huyeng/openapi-action> (besucht am 07. 06. 2022).
- [19] buildingSMART, *BCF REST API*, 13. Feb. 2022. Adresse: <https://github.com/buildingSMART/BCF-API> (besucht am 09. 05. 2022).