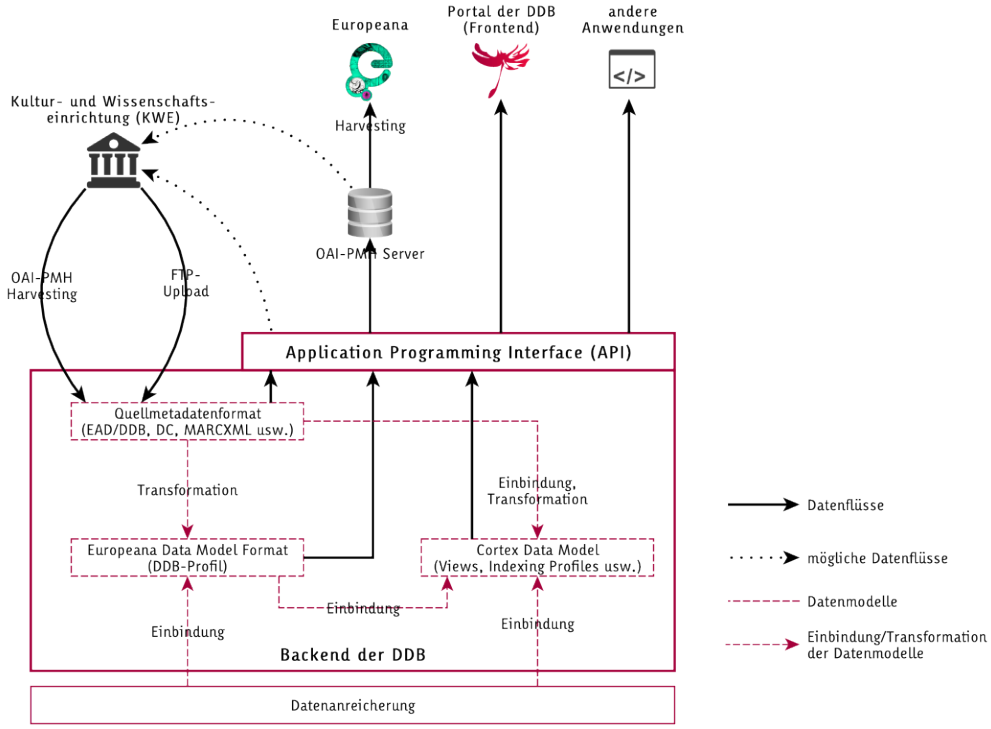
Programmierschnittstelle

Application Programming Interface

Das **A**pplication **P**rogramming **I**nterface (API) der Deutschen Digitalen Bibliothek bietet über Methoden Zugriff auf die Daten des Backends (Cortex). Ziel der Programmierschnittstelle ist die Entwicklung vielfältiger Anwendungen, die die Daten und Funktionalitäten der DDB nutzen. Dazu gehören externe Webseiten, Anwendungen und Apps, die eine domainspezifische Verarbeitung und Darstellung der Daten der DDB ermöglichen, auch Daten anreichern können und somit das Prinzip von Linked Open Data unterstützen. Das API wird als RESTful Web Service über das **H**yper**t**ext **T**ransfer **P**rotocol (HTTP) angeboten und kann über Query- und Path-Parameter sowie dem HTTP-Request-Header gesteuert werden.

Datenflüsse und -modelle



RESTful Web Service

Für öffentliche APIs hat sich in den vergangenen Jahren ein auf dem REST-Prinzip basierender Standard herausgebildet, dem auch die API der Deutschen Digitalen Bibliothek (DDB) folgt. Die „gemeinsame Sprache“, in der die Anwendung, die auf das API zugreift und der Server, der das API bereitstellt, sich miteinander unterhalten, ist das Hypertext Transfer Protokoll HTTP, auf dem das World Wide Web aufbaut.

Für den Zugriff auf die Daten des Backends, also der Datenbank der Deutschen Digitalen Bibliothek wird eine Java API for RESTful Web Services angeboten. Diese stellt eine universelle und plattformunabhängige Schnittstelle dar, deren Abfragen über Path/Query-Parameter und dem HTTP-Header gesteuert werden kann. Der öffentliche Teil des API umfasst 15 Methoden. Für die Nutzung des API ist ein Entwicklerschlüssel notwendig, der wiederum ein Benutzerkonto im DDB-Portal voraussetzt. Zusätzlich zu den Allgemeinen Nutzungsbedingungen gibt es speziell für das API eigene API-Nutzungsbedingungen. Diesen Nutzungsbedingungen muss der interessierte Entwickler im Laufe des Registrierungsprozesses zustimmen.

Query- und Path-Parameter

Analog zum Webbrowser, der eine Webseite anzeigen will und zu diesem Zweck eine Nachricht mit der zugehörigen URL per HTTP an den Webserver schickt, sendet die Anwendung, die auf das Backend der DDB zugreifen möchte, per HTTP, an einen durch eine URL eindeutig beschriebenen Endpunkt, eine Nachricht. Diese URL ist nach einem bestimmten Muster strukturiert, z. B.:

```
https://api.deutsche-digitale-bibliothek.de/search?<Query-Parameter>
```

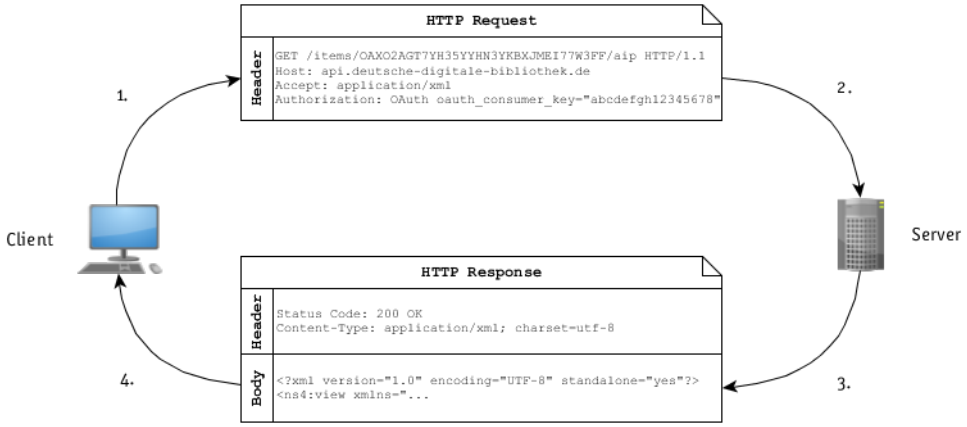
Ganz vorne steht das verwendete Protokoll. Darauf folgt der Hostname, sozusagen die menschenlesbare Adresse des Servers, der das API ausliefert, woran sich der Pfad (Path) zu der Methode anschließt, die zu dem Endpunkt gehört. Mit einem Fragezeichen kann ein Satz sogenannter **Query-Parameter** angeschlossen werden, z. B.:

```
https://api.deutsche-digitale-bibliothek.de/search?query=Egmo*&offset=21&rows=2
```

Neben den Query Parametern sind die **Path Parameter** zu nennen. Diese spezifizieren nicht nur die aufzurufende Methode (wie bereits oben eingeführt), sondern können auch weitere Daten transportieren. So z. B. die ID des angefragten Datensatzes:

```
https://api.deutsche-digitale-bibliothek.de/items/OAXO2AGT7YH35YYHN3YKBXJMEI77W3FF/edm
```

HTTP-Request und -Response



Zu der gemeinsamen Sprache HTTP, über die das API angesprochen wird, gehört eine Reihe von Verben, von denen für den öffentlichen Teil des APIs der DDB nur GET eine Rolle spielt, da über den öffentlichen Teil Daten nur gelesen, aber nicht geschrieben werden können. Die per HTTP ausgetauschten Nachrichtenpakete haben ein bestimmtes Format, welches in der Spezifikation von HTTP definiert ist.

Diese Nachrichtenpakete bestehen aus einem Kopf (Header) und einem Körper (**Body**). Bei GET-Anfragen bleibt der HTTP Body immer leer, da keine Daten zum Server transportiert werden. Die Antwort des Servers enthält hingegen im Body die angefragten Daten, typischerweise in einem strukturierten Datenformat wie XML oder JSON. Weiterhin werden sowohl im Kopf der Anfrage (**Request**) als auch im Kopf der Antwort (**Response**) weitere Angaben übertragen, die z. B. das angeforderte/akzeptierte Datenformat oder die Sicherheit (Authentifizierung) betreffen.

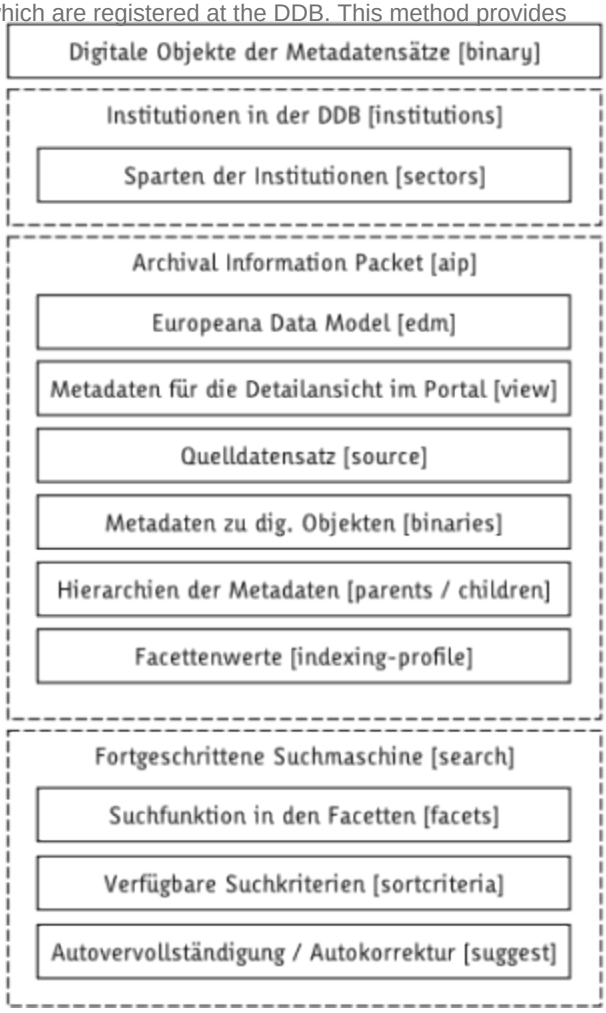
Am nebenstehenden Bild soll dieser Prozess veranschaulicht werden: Der Client, also der Computer bzw. die Anwendung die Daten vom Backend der DDB haben möchte, sendet einen **HTTP Request** an den Server. Dazu muss zunächst ein gültiger HTTP Request Header formuliert werden (1), in dem die angeforderte Ressource spezifiziert wird, sich der Client gegenüber dem Server Authentifiziert (OAuth) und das in der Antwort akzeptierte Format der Daten festgelegt wird. Diese Anfrage wird zum Server gesandt (2) und vom Server mit einer Antwort dem **HTTP Response** (3) beantwortet. Der Status Code im HTTP Response Header gibt anschließend für den Client Aufschluss darüber, ob eine Anfrage erfolgreich war. Eine mit dem Wert 200 quittierte Anfrage wurde erfolgreich ausgeführt und enthält im Body des HTTP Response die angeforderten Daten, die durch den Client verarbeitet werden können (4).

API-Methoden

Der Aufbau der Methoden der API orientiert sich an dem Datenmodell der Deutschen Digitalen Bibliothek. Das **A**rchival **I**nformation **P**acket (AIP) kapselt die Informationen der Bestandteile EDM, View, Source, Binaries, Parents, Children und Indexing-Profile. Mit Hilfe der Methode `search` kann auf die auf Apache Solr aufbauende Suchmaschine zugegriffen werden. Binärobjekt wie Vorschaubilder stellt die Methode `binary` zur Verfügung.

- binary — The method binary returns the content of a binary file of an item for a given item-ID. This method provides response data as application/octet-stream. A binary file at DDB can be a picture, a tumbnail of a picture, a video clip, an audio file etc. It is a read-only service and must be accessed with a HTTP-GET-request.
- entities — The method entities is providing access to the the Lucene search index of entities used at DDB. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.

- institutions — The method institutions returns a list of institutions which are registered at the DDB. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.
    - sectors — The method sectors returns the list of available institution sectors at the DDB. Each sector contains a name in the property value and the number of institutions that belong to this sector (count). As institutions can belong to multiple sectors, the overall sum of the different counts can be higher than the total number of institutions. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.
- items
    - aip — The method aip returns the Archive Information Package (AIP) of an item for a given item-ID. An AIP contains all available information of an item including a persistent identifier. This method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
    - binaries — The method binaries returns a list of binary data files related to an item for a given item-ID. Binary data can be accessed with the binary method. The binaries method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
    - children — The method children returns metadata of these items which are related to the inquired item and which are one level deeper (child item) in the hierarchy than the inquired item. The child items will be sorted according to the position field of the hierarchy nodes. If the position is the same the label will be used for sorting. The provided metadata can be empty if the item does not have any child items. This method provides response data as application/json and application/xml. It is a read-only s
    - edm — The method edm returns the Europeana Data Model (DDB profile) of an item for a given item-ID. This method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
    - indexing-profile — The method indexing-profile returns the profile of an item for a given item-ID, which was used for the indexing process. This method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
    - parents — The method parents returns item-IDs of these items which are related to the inquired item and which are one or more levels higher (parent, grandparent ... item) in the hierarchy than the inquired item. This means that all parent items up to the root item will be provided. The provided metadata can be empty if the item does not have any parent items. This method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
    - source — The method source returns the initial XML metadata of an item for a given item-ID. The format can be one of the accepted input data formats (e.g. MARCXML, METS/MODS, LIDO, Dublin Core). This method provides response data only as application/xml because the DDB only accept XML-based metadata from its contributing institutions. It is a read-only service and must be accessed with a HTTP-GET-request.
    - view — The method view returns the view of an item for a given item-ID. A view is the data set a frontend page at DDB is based on. This method provides response data as application/json and application/xml. It is a read-only service and must be accessed with a HTTP-GET-request.
- search — The method search is providing an interface to the search engine of the DDB. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request. dd
    - facets — The method facets of search returns all available facets or all available facets of a specific type. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.
    - organization — The method search/organization is providing access to the the Lucene search index of organizations used at DDB. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.
    - sortcriteria — The method sortcriteria of search returns the available sort criteria and the default criterion of the



search result sets. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.

- suggest — The method suggest of search returns suggestions for a given query. This method provides response data only as application/json. It is a read-only service and must be accessed with a HTTP-GET-request.