




SOFTWARE TOOL ARTICLE

ENA Source Attribute Helper: An Application Programming Interface to facilitate accurate reference to biological source data [version 1; peer review: awaiting peer review]

Vikas Gupta, Joana Paupério , Josephine Burgin, Suran Jayathilaka, Guy Cochrane

European Molecular Biology Laboratory, European Bioinformatics Institute, Wellcome Genome Campus, Hinxton, CB10 1SD, UK

V1 First published: 13 Sep 2022, 11:1042
<https://doi.org/10.12688/f1000research.123934.1>

Latest published: 13 Sep 2022, 11:1042
<https://doi.org/10.12688/f1000research.123934.1>

Abstract

Background: Metadata attributes of sequences that accurately reference their biological sources, as specimens or other materials of origin, and link with natural history collections, are essential to facilitate the connections between different fields in life sciences and promote reusability of data. However, metadata used to reference the biological source of sequences available within the molecular data repositories are not always well structured or comprehensive.

Methods: Within the scope of the Horizon 2020 project Biodiversity Community Integrated Knowledge Library (BiCIKL), we have developed a tool, the European Nucleotide Archive (ENA) Source Attribute Helper Application Programming Interface (API), to help users accurately report biological source-related sequence and sample attributes. This tool currently focuses on the attributes in which specimens, cultures or other materials are identified, from which the sequence data were derived, and uses curated data to obtain the unique codes for the institutions and collections holding the vouchers. The API's main functions include the presentation of metadata associated with queried institutions or collections, validation of institution and collection codes in the attribute strings provided by the user, and the construction of an attribute string based on user-entered data. The API does not however support the search of voucher specimen codes, as these need to be obtained directly from the voucher institutions. We describe the API and discuss use cases for its different endpoints. The API is available at <https://www.ebi.ac.uk/ena/sah/api/>.

Conclusions: We expect the API to promote and support the initial submission and any subsequent curation of biological source attributes, and hereby contribute to better links between sequence data and natural history collections, and hence on to taxonomy and biodiversity research, towards increasing the discoverability, reusability and impact of data.

Open Peer Review

Approval Status *AWAITING PEER REVIEW*

Any reports and responses or comments on the article can be found at the end of the article.

Keywords

European Nucleotide Archive, submission tools, validation functions, specimen voucher, culture collection, bio material, NCBI Biocollections



This article is included in the **Bioinformatics** gateway.

EMBL-EBI



This article is included in the **EMBL-EBI** collection.

Corresponding author: Joana Paupério (joanap@ebi.ac.uk)

Author roles: **Gupta V:** Software, Writing – Original Draft Preparation, Writing – Review & Editing; **Paupério J:** Writing – Original Draft Preparation, Writing – Review & Editing; **Burgin J:** Writing – Review & Editing; **Jayathilaka S:** Software, Writing – Review & Editing; **Cochrane G:** Conceptualization, Supervision, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This research was financially supported by the European Union's Horizon 2020 research and innovation programme under the grant agreement No 101007492 (project name BiCIKL).

Copyright: © 2022 Gupta V *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Gupta V, Paupério J, Burgin J *et al.* **ENA Source Attribute Helper: An Application Programming Interface to facilitate accurate reference to biological source data [version 1; peer review: awaiting peer review]** F1000Research 2022, 11:1042 <https://doi.org/10.12688/f1000research.123934.1>

First published: 13 Sep 2022, 11:1042 <https://doi.org/10.12688/f1000research.123934.1>

Introduction

The generation and archiving of sequence data and associated metadata at large scale have transformed and promoted research in the life sciences. Sequence data have been essential to scientific breakthroughs in several fields such as medicine, food security, evolutionary biology and biodiversity conservation.

One of the most important aspects of the archiving of sequence data is metadata management, which is crucial for the accurate description of Earth's genetic and genomic biodiversity and its preservation in molecular sequence collections (Waterhouse *et al.* 2021). By holding enriched metadata, such as biological source attributes, that describe the material provenance of sequence data (allowing linking to the specimen of origin), molecular sequence collections facilitate connections between molecular biology, taxonomy, systematics and biodiversity research, increasing the discoverability and usability of data by researchers worldwide.

The infrastructure for storing and sharing of sequence data is the International Nucleotide Sequence Database Collaboration (INSDC, Arita *et al.* 2021) that operates between the DNA Data Bank of Japan (DDBJ, Fukuda *et al.* 2021), the National Centre for Biotechnology Information (NCBI, Sayers *et al.* 2021) and the European Nucleotide Archive (ENA, Cummins *et al.* 2022) that stands as its European node. The INSDC contains currently over 236 million sequences and 1.7 billion reads (<https://www.ncbi.nlm.nih.gov/genbank/statistics/>), holding a large body of associated metadata related to sequenced sample sources, as culture collection or natural history collection annotations. However, for a number of records these metadata may be incomplete or ambiguous, which hinders the linking of the sequence data to their origin and therefore reduces data reusability.

The Biodiversity Community Integrated Knowledge Library (BiCIKL) is a Horizon 2020 project that aims to establish open science practices in the biodiversity domain by providing Findable, Accessible, Interoperable and Reusable (FAIR) access and developing new methods and workflows for linking data along the biodiversity research cycle, namely from data resources of molecular biology, natural history collections, taxonomy, and literature (Penev *et al.* 2022). To take full advantage of these workflows a foundation of well-structured and accessible metadata is required, namely in the molecular sequence databases. Therefore, in the scope of this project we have developed a tool for driving the accurate and complete reporting of biological source metadata: the ENA Source Attribute Helper.

The metadata that refers to the biological source of sequence data is described for sequences in the source feature qualifiers that are embedded in the sequence flat files (INSDC 2021, <https://www.ebi.ac.uk/ena/WebFeat/>) and in the samples' attributes. These will be hereafter referred simply as 'attributes'. These attributes are submitted to the ENA at the time of data deposition or in subsequent updates. ENA holds many routes for data submission (such as web interfaces, RESTful Application Programming Interfaces (APIs) and locally installed command-line tools), so there is no single point of entry for supporting submission of accurate provenance metadata. Therefore, the Source Attribute Helper is a publicly accessible open-source tool that may be used as a free-standing service independently across platforms and workflows. The initial version of the tool focuses on the sequence and sample attributes that identify the specimen, culture, or material from which the sequence was derived, namely /specimen_voucher, /culture_collection, and /bio_material. These attributes are formatted according to the Darwin Core Standards (Wieczorek *et al.* 2012) and follow a Darwin Core Triplet format, composed of Institution code, collection code and the specimen, culture, or material id, accordingly (Table 1). The tool was developed to help users fetch accurate information regarding the institution and collections codes of the specimen, culture, or material, and construct and validate the string to be submitted as an attribute

Table 1. Attributes for the biological source of sequence data addressed in the current version of the ENA Source Attribute Helper API. The value of the attributes follows a Darwin Core triplet format (according to Darwin Core Standards, Wieczorek *et al.* 2012). Specimen ID, culture ID, and material ID are mandatory values. Institution code is optional for specimen voucher and bio material, but mandatory for culture collection. Collection code is always optional. When collection code is provided, institution code is mandatory (see INSDC 2021, also available at <https://www.ebi.ac.uk/ena/WebFeat/>, for more details on these attributes). ENA, European Nucleotide Archive; API, Application Programming Interface; INSDC, International Nucleotide Sequence Database Collaboration.

Attribute	Definition	Value format
Specimen voucher	Identifier for the specimen from which the data was obtained	[<institution-code>:<collection-code>:]<specimen_id>
Culture collection	Identifier for the culture from which the data was obtained	<institution-code>:<collection-code>:<culture_id>
Bio material	Identifier for the biological material from which the data was obtained	[<institution-code>:<collection-code>:]<material_id>

of the sequence data. The tool does not however support the search of voucher specimen codes, as these need to be obtained directly from the institutions.

In this paper we describe the design and implementation of the API. We also describe use cases for its application, highlighting its utility for increasing the accuracy of biological source attributes in molecular databases and promoting reusability.

Methods

The ENA Source Attribute Helper, although described as a single tool, comprises several endpoints with different functions, namely, to display metadata associated with institutions or collections, to validate the attribute string provided by the user according to the institutions and collections database, and to construct the attribute string based on data input by the user. The code is available from [GitHub](#) and is archived with [Zenodo](#) (Jayathilaka & Gupta 2022).

For the retrieval of information on the institutions and collections and subsequent validation, the application uses the data available in the [NCBI Biocollections](#) (RRID:SCR_016459). NCBI Biocollections is a curated database of metadata for herbaria, museums, culture collections, and other natural history collections, that are connected to records in INSDC, and is maintained by the NCBI taxonomy group (Sharma *et al.* 2018). It includes institution and collection codes and their URLs, where available, that allow users to find additional information. New records are added to the database upon submission of information together with sequence records to INSDC. The ENA Source Attribute Helper consumes the curated data from NCBI Biocollections to get, validate and construct the values for the attributes (Tables 2 and 3). Currently, the available Biocollections database files are retrieved manually from the ftp server and imported into the ENA ElasticSearch datastore.

The development of the ENA Source Attribute Helper API was based on the following tools and frameworks:

1. **Spring Boot API framework:** This is a framework used for building RESTful APIs that are accessible from various platforms/clients including but not limited to web browsers, mobile devices, desktop applications *etc.*
2. **ElasticSearch** datastore: The application utilises strengths of ElasticSearch datastore to enable text search over JavaScript Object Notation (JSON) data over multiple properties and provide suggestions/similar matches.

Table 2. Description of fields included in the NCBI Biocollections institutions database imported into the ENA ElasticSearch datastore. NCBI, National Centre for Biotechnology Information; ENA, European Nucleotide Archive.

Field	Example	Type	Requirement	Notes
_id	x-P-TIEBpiSBteIppVU0	System set uuid	System set	System set uuid.
inst_id	1111	Integer	Mandatory	Provided Institution id
inst_code	CAMZM	String	Mandatory	Institution Code
unique_name	UMZC	String	Mandatory	Institution Unique Name
synonyms	AEIC	String	Optional	Institute Synonym
inst_name	University Museum of Zoology Cambridge	String	Mandatory	Name of the Institution
country	United Kingdom	String	Mandatory	Institution Country
address	Downing Street, Cambridge, CB2 3EJ, Cambridge	String	Mandatory	Address of the Institution
collection_type	museum	String	Mandatory	Type of Collection
qualifier_type	specimen_voucher	String	Mandatory	Attribute: specimen_voucher/bio_material/culture_collection
home_url	http://www.zoo.cam.ac.uk/museum/	String	Optional	Home Page URL
url_rule		String	Optional	URL Rule Page

Table 3. Description of fields included in the NCBI Biollections collections database imported into the ENA ElasticSearch datastore. NCBI, National Centre for Biotechnology Information; ENA, European Nucleotide Archive.

Field	Example	Type	Requirement	Notes
_id	dee11d4e-63c6-4d90-983c-5c9f1e79e96c	System set uuid	System set	System set uuid
coll_id	222	Integer	Mandatory	Provided Collection id
inst_id	12345	Integer	Mandatory	Mapped Institution id
coll_code	Annelid	String	Mandatory	Collection Code
coll_name	Annelid Collection	String	Mandatory	Name of the Collection
coll_type	museum	String	Mandatory	Type of Collection
qualifier_type	specimen_voucher	String	Mandatory	Attribute: specimen_voucher/ bio_material/ culture_collection
coll_url	http://nature.ca/collections/inverts_e.cfm	String	Optional	Collection Reference URL
coll_url_rule	https://science.mnhn.fr/institution/mnhn/collection/ar/item/ar	String	Optional	Collection URL Rule

3. **Spring Data** and object–relational mapping (ORM): The application utilises Spring Data libraries to create abstractions over repository and custom object mappings along with usage of an ORM (like **Hibernate**).
4. **Postman**: This is an open-source tool for testing, monitoring and publishing APIs.
5. **Swagger**: This is an Interface Description Language (IDL) used for the description of RESTful APIs. It allows the visualisation of the various API endpoints and an easy execution of the commands.

Implementation

Construct & validation flows

The main function of the API is to validate and construct the Attribute Values (see [Table 1](#)), to ensure that these are aligned with the format definition. The user inputs a code or name (at least one character) for the institution or collection and the application suggests the closest options available, so that the user can select the correct option. The construct and validation flows are represented in [Figure 1](#).

Core

The core of the application is built using the Spring Boot framework, which follows a layered architecture approach (Presentation Layer, Business Layer, Persistence Layer, Database Layer) in which each layer communicates to other layers in a hierarchical order. The Database entities (ElasticSearch datastore) and the Backend APIs (Web API Layer) are represented in [Figure 2](#).

Operation

API endpoints

API endpoints are the channels through which other applications can communicate with or consume an API. They are represented by Universal Resource Locators (URL), which serve as points of entry.

[Table 4](#) describes the various endpoints that this API provides. The endpoint to get error-codes is an additional endpoint that allows users to fetch the definitions of the error codes, which may be returned by the system. This may be useful for system integration and error handling on the client side.

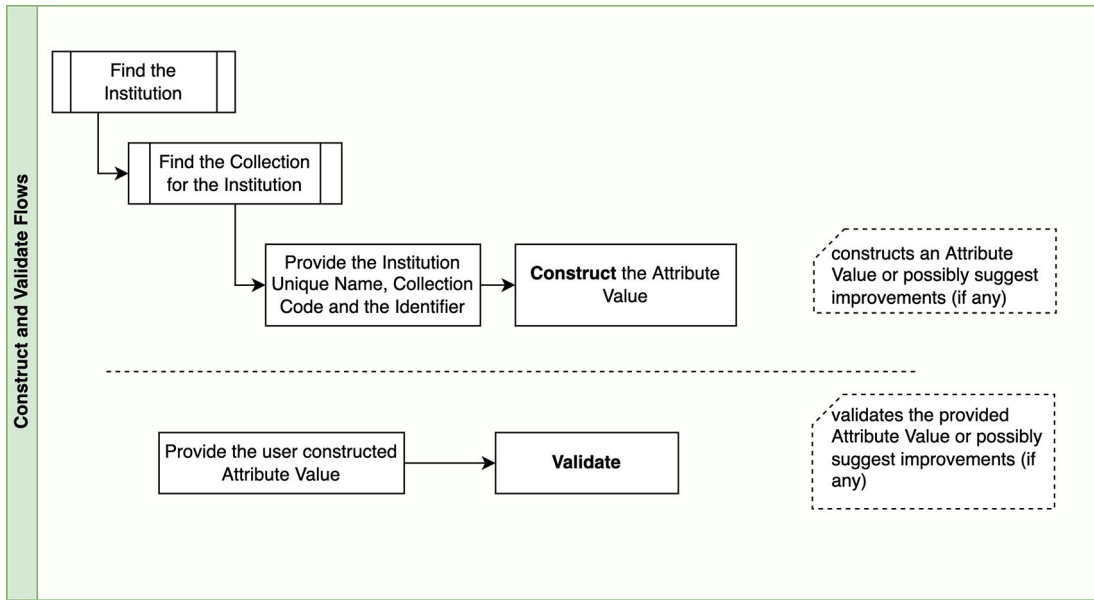


Figure 1. Construct & validation flow diagram.

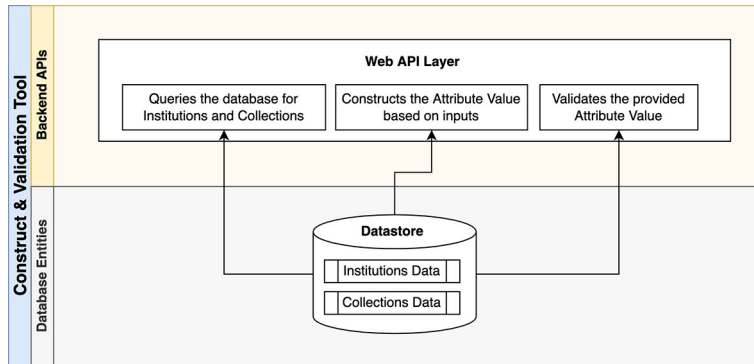


Figure 2. Construct & validation tool components of the application. API, Application Programming Interface.

API access & tools

The API can be accessed using:

- any Web Browser
- any scripting/programming language-based REST client
- command line tools like cURL and Wget
- testing tools like Swagger user interface (UI) and Postman

Testing tools, such as the Swagger UI, facilitate the usage of the API even by a non-technical person (Figure 3), while other tools such as Postman may require a higher level of technical understanding to know how to consume the API.

Using the Swagger interface

Swagger is a web browser based graphical UI that provides a set of form fields and hints for interacting with a RESTful API. It helps a user to interact with and test the API by hiding the complexity of building correct requests. It uses

Table 4. ENA Source Attribute Helper API Endpoints description and success and failure responses. ENA, European Nucleotide Archive; API, Application Programming Interface.

API Endpoint	Verb	Action	Success	Failure
/institution/{ivalue}	GET	Finds Institution using institution name or code. If the institution name or code is not fully known, 1 or more characters can be provided. API searches both for exact matches and for partial matches by either institution name or institution code.	200 OK	400 Bad Request
/institution/{institutionUniqueName}/collection	GET	Gets all collections by institution unique name	200 OK	400 Bad Request
/institution/{institutionUniqueName}/collection/{cvalue}	GET	Gets collection by institution's unique name and collection code. If the collection name or code is not fully known, 1 or more characters can be provided.	200 OK	400 Bad Request
/validate	GET	Validates the provided attribute string	200 OK	400 Bad Request
/construct	GET	Constructs the attribute string	200 OK	400 Bad Request
/error-codes	GET	Gets the error codes definition	200 OK	400 Bad Request

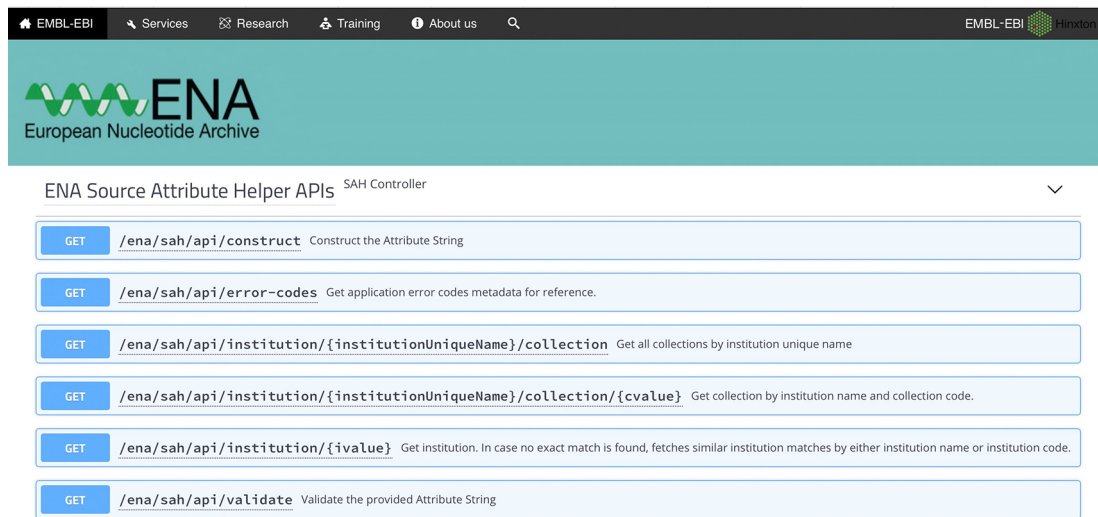


Figure 3. Swagger UI for the ENA Source Attribute Helper API. UI, User Interface; ENA, European Nucleotide Archive; API, Application Programming Interface.

annotations and descriptions in the source code of the application to describe the API in human readable format. [Figure 3](#) shows the general UI for Swagger, and an example of a JSON response for the /validate endpoint (which validates the given attribute string) is displayed in [Figure 4](#). [Figure 5](#) shows an example of a JSON response for the /construct endpoint, which validates input and constructs the attribute string based on the provided parameters.

Using the cURL command-line tool

cURL is a widely available free and open-source command-line tool for transferring data using URL syntax. [Figures 6](#) and [7](#) show two examples of the usage of the /validate and /construct API endpoints to validate and construct the attribute string, respectively, using cURL.

The image shows the Swagger UI for the /validate endpoint. At the top, it indicates a GET request to /ena/sah/api/validate. The description is "Validate the provided Attribute String". Under Parameters, there is a table with columns "Name" and "Description". The first row is for "qualifier_type" with description "Acceptable values are (specimen_voucher, bio_material, culture_collection)" and an "Add Item" button. The second row is for "value" (required) with description "qualifier string is required" and a text input field containing "MSNT:FAZC:123456". Below the parameters are "Execute" and "Clear" buttons. The Responses section shows a 200 status with "Response content type" set to "*/*". The Curl section shows the command: curl -X GET "https://wwwdev.ebi.ac.uk/ena/sah/api/validate?value=MSNT%3AFAZC%3A123456" -H "accept: */*". The Request URL is https://wwwdev.ebi.ac.uk/ena/sah/api/validate?value=MSNT%3AFAZC%3A123456. The Server response section shows a 200 status and a JSON response body: {"success": true, "matchLevel": "Exact", "timestamp": "2022-08-03T15:54:27.388", "inputValue": "MSNT:FAZC:123456", "matches": [{"match": "MSNT<ITA-Torino>:FAZC:123456", "institution": {"institutionCode": "MSNT", "uniqueName": "MSNT<ITA-Torino>", "institutionName": "Museo Regionale di Scienze Naturali, Torino", "country": "Italy", "address": "Torino", "qualifierType": ["specimen_voucher"], "homeUrl": ""}, "collections": [{"collectionCode": "FAZC"}]}]}.

Figure 4. Swagger UI for /validate endpoint. UI, User Interface.

The image shows the Swagger UI for the /construct endpoint. At the top, it indicates a GET request to /ena/sah/api/construct. The description is "Construct the Attribute String". Under Parameters, there is a table with columns "Name" and "Description". The first row is for "collection" with description "a valid collection code is required" and a text input field containing "FAZC". The second row is for "id" (required) with description "an identifier is required" and a text input field containing "123456". The third row is for "institution" (required) with description "a valid institution unique name is required" and a text input field containing "MSNT". The fourth row is for "qualifier_type" with description "Acceptable values are (specimen_voucher, bio_material, culture_collection)" and an "Add Item" button. Below the parameters are "Execute" and "Clear" buttons. The Responses section shows a 200 status with "Response content type" set to "*/*". The Curl section shows the command: curl -X GET "https://wwwdev.ebi.ac.uk/ena/sah/api/construct?collection=FAZC&id=123456&institution=MSNT" -H "accept: */*". The Request URL is https://wwwdev.ebi.ac.uk/ena/sah/api/construct?collection=FAZC&id=123456&institution=MSNT. The Server response section shows a 200 status and a JSON response body: {"success": true, "matchLevel": "Exact", "timestamp": "2022-08-03T15:58:39.018", "inputValue": "MSNT:FAZC:123456", "matches": [{"match": "MSNT<ITA-Torino>:FAZC:123456", "institution": {"institutionCode": "MSNT", "uniqueName": "MSNT<ITA-Torino>", "institutionName": "Museo Regionale di Scienze Naturali, Torino", "country": "Italy", "address": "Torino", "qualifierType": ["specimen_voucher"]}}]}.

Figure 5. Swagger UI for /construct endpoint. UI, User Interface.


```

| $curl -s -X GET 'https://wwwdev.ebi.ac.uk/ena/sah/api/validate?value=MSNT:FAZC:123456&qualifier_type=specimen_voucher' |
{
  "success": true,
  "matchLevel": "Exact",
  "timestamp": "2022-08-03T16:02:28.366",
  "inputValue": "MSNT:FAZC:123456",
  "matches": [
    {
      "match": "MSNT<ITA-Torino>:FAZC:123456",
      "institution": {
        "institutionCode": "MSNT",
        "uniqueName": "MSNT<ITA-Torino>",
        "institutionName": "Museo Regionale di Scienze Naturali, Torino",
        "country": "Italy",
        "address": "Torino",
        "qualifierType": [
          "specimen_voucher"
        ],
        "homeUrl": "",
        "collections": [
          {
            "collectionCode": "FAZC",
            "collectionName": "Franco Andreone Zoological Collection",
            "qualifierType": [
              "specimen_voucher"
            ],
            "collectionUrl": ""
          }
        ]
      }
    }
  ]
}

```

Figure 6. cURL request for /validate API endpoint. API, Application Programming Interface.

```

$curl -s -X GET 'https://wwwdev.ebi.ac.uk/ena/sah/api/construct?institution=HSUV&collection=Bird&id=123456' | jq
{
  "success": true,
  "matchLevel": "Partial",
  "timestamp": "2022-08-03T16:05:36.539",
  "message": "Multiple similar matches found. Please select the appropriate one or retry the search",
  "inputValue": "HSUV:Bird:123456",
  "matches": [
    {
      "match": "HSUVM:Bird:123456",
      "institution": {
        "institutionCode": "HSUVM",
        "uniqueName": "HSUVM",
        "institutionName": "Humboldt State University Vertebrate Museum",
        "country": "USA",
        "address": "1 Harpst St. Arcata California 95521",
        "qualifierType": [
          "specimen_voucher"
        ],
        "homeUrl": "http://www.humboldt.edu/vmuseum/",
        "collections": [
          {
            "collectionCode": "Bird",
            "collectionName": "Bird Collection",
            "qualifierType": [
              "specimen_voucher"
            ],
            "collectionUrl": ""
          }
        ]
      }
    },
    {
      "match": "HSU<USA-CA>:Birds:123456",
      "institution": {
        "institutionCode": "HSU",
        "uniqueName": "HSU<USA-CA>",
        "institutionName": "Humboldt State University",
        "country": "USA",
        "address": "Vertebrate Museum Humboldt State University 1 Harpst Street Arcata, CA 95521",
        "qualifierType": [
          "specimen_voucher"
        ]
      }
    }
  ]
}

```

Figure 7. cURL request for /construct API endpoint. API, Application Programming Interface.

Using the Postman API client

The **Postman** API client is a tool to easily explore, debug, and test APIs while also enabling users to define complex API requests for HTTP, REST, SOAP, GraphQL, and WebSockets. In the development of the tool, we engaged it to inspect API endpoints and their responses. An example JSON response for the /validate endpoint, which validates a given attribute string and presents the support data for the institution and collection values, is shown in [Figure 8](#).

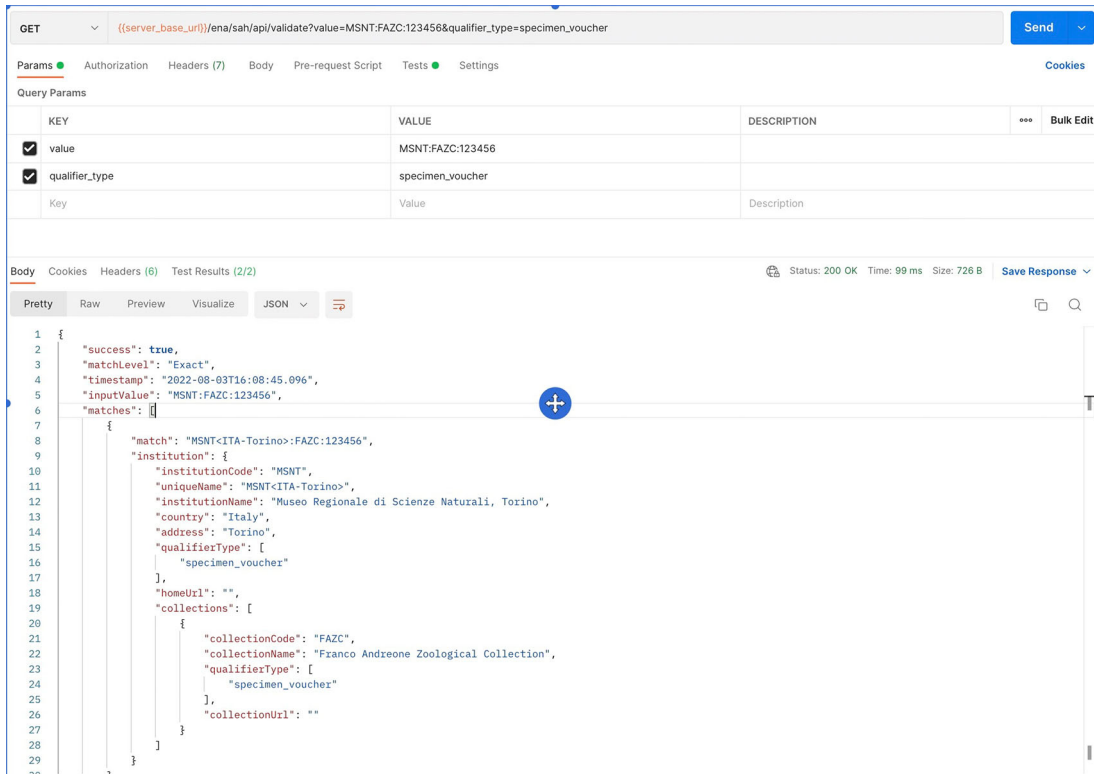


Figure 8. Postman UI for /validate endpoint, showing the request and the obtained response. UI, User Interface.

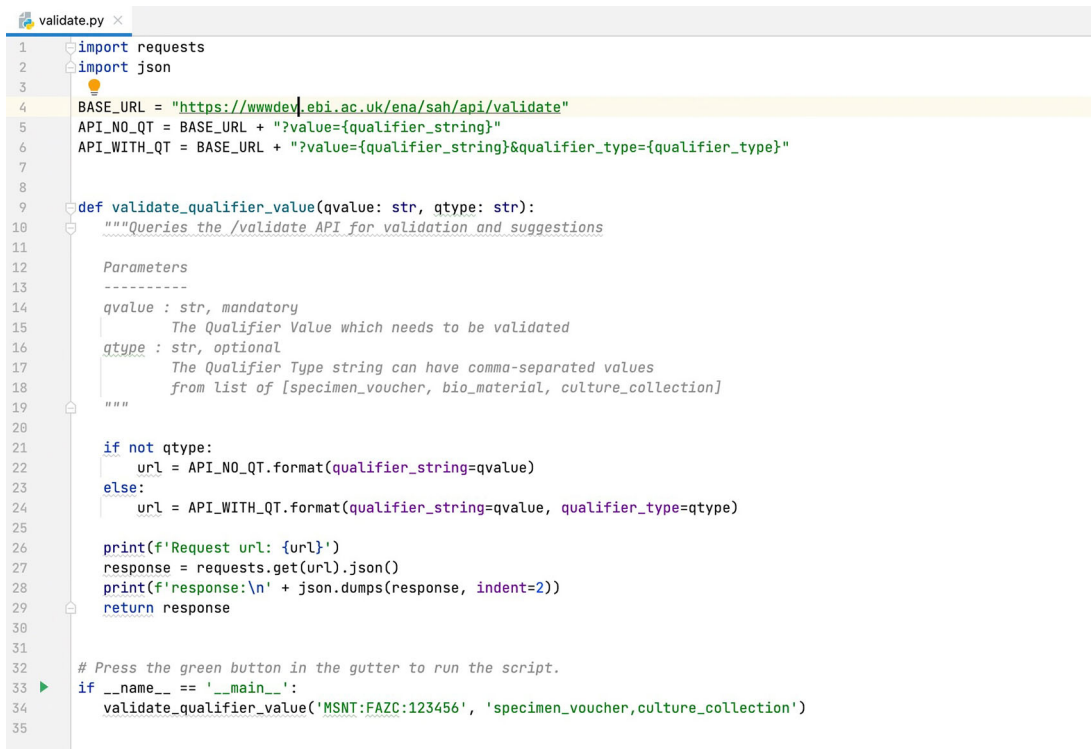


Figure 9. Python sample code for /validate API endpoint. API, Application Programming Interface.

```

1  import requests
2  import json
3
4  BASE_URL = "https://wwwdev.ebi.ac.uk/ena/sah/api/construct"
5  API_NO_QT = BASE_URL + "?institution=HSUV&collection=Bird&id=123456"
6  API_WITH_QT = BASE_URL + "?institution={inst}&collection={coll}&id={id}&qualifier_type={qualifier_type}"
7
8
9  def validate_qualifier_value(institutionUniqueName: str, collectionCode: str, identifier: str, qtype: str):
10     """Queries the /construct API endpoint for building qualifier string and suggestions in case of partial matches
11
12     Parameters
13     -----
14     institutionUniqueName : str, mandatory
15         The institution unique name
16     collectionCode : str, mandatory
17         The collection code
18     identifier : str, mandatory
19         The identifier value
20     qtype : str, optional
21         The Qualifier Type string can have comma-separated values
22         from list of [specimen_voucher, bio_material, culture_collection]
23     """
24     if not qtype:
25         url = API_NO_QT.format(inst=institutionUniqueName, coll=collectionCode, id=identifier)
26     else:
27         url = API_WITH_QT.format(inst=institutionUniqueName, coll=collectionCode, id=identifier, qualifier_type=qtype)
28     print(f'Request url: {url}')
29     response = requests.get(url).json()
30     print(f'response:\n' + json.dumps(response, indent=2))
31     return response
32
33
34 # Press the green button in the gutter to run the script.
35 if __name__ == '__main__':
36     response = validate_qualifier_value('HSUV', 'Bird', '123456', 'specimen_voucher,culture_collection')
37

```

Figure 10. Python sample code for /construct API endpoint. API, Application Programming Interface.

Other API clients - Python

Python is a scripting/programming language that allows a quick output and integrates systems more effectively. Figures 9 and 10 show basic code examples to demonstrate querying two of the available API endpoints - /validate and /construct.

Deployment

The ENA Source Attribute Helper application is deployed on the European Molecular Biology Laboratory's European Bioinformatics Institute (EMBL-EBI) infrastructure, and loads balanced on a cluster of servers for resiliency and high availability.

Use cases

Institution codes

Users submitting sequence related data may need to look for the unique code for the institution holding the voucher associated with the data. The API endpoint *Get Institution* allows the user to fetch the Institution details by providing either the institution name or code (Table 5). The type of attribute ('qualifier_type': specimen voucher, culture collection, or bio material) may also be optionally specified, but if none is provided the API will search within all attributes. The API searches both for exact matches and for partial matches. The API response will include the metadata for all institutions with exact or partial matches to the input value, allowing the user to confirm the details of the institution that is holding the voucher.

Table 5. Parameters required for the API endpoint *Get Institution*. Requirements and an example are also provided. API, Application Programming Interface.

Parameters	Example	Type	Requirement	Notes
ivalue	CAMZX	String	Mandatory	Institution name/code - to search for institution(s)
qualifier_type	specimen_voucher, bio_material, culture_collection	String	Optional	Filters results for a specific attribute

Table 6. Parameters required for the API endpoint *Get Collections for the Institution*. Requirements and an example are also provided. API, Application Programming Interface.

Parameters	Example	Type	Requirement	Notes
institutionUniqueName	CAMZX	String	Mandatory	Institution's unique name
qualifier_type	specimen_voucher, bio_material, culture_collection	String	Optional	Filters results for a specific attribute

Table 7. Parameters required for the API endpoint *Get Collections by Institution Unique name and Collection Code*. Requirements and an example are also provided. API, Application Programming Interface.

Parameters	Example	Type	Requirement	Notes
institutionUniqueName	CAMZX	String	Mandatory	Institution's unique name
cvalue	herp	String	Mandatory	Collection Code
qualifier_type	specimen_voucher, bio_material, culture_collection	String	Optional	Filters results for a specific attribute

Collection codes

Once the user knows the unique institution code for their voucher, they may need to identify the unique code for the collection to input in the attribute string. The API has two endpoints that allow users to search for the collection codes. In both endpoints the type of attribute ('qualifier_type': specimen voucher, culture collection, or bio material) may be optionally specified.

The endpoint *Get Collections for the Institution* allows to fetch all collections in a given institution by providing the institution's unique name (unique code, [Table 6](#)). This operation looks only for an exact match of the institution's unique name and returns the complete list of collections within that institution and associated metadata. If the institution's unique name is not found in the database the endpoint does not return any record.

The endpoint *Get Collections by Institution Unique name and Collection Code* allows users to obtain the metadata of a given collection of an institution by providing the institution's unique name (unique code) and known collection code ([Table 7](#)). This endpoint searches for an exact match of the institution's unique name, and a full or partial match of the collection code and returns the metadata for the collection found. If the institution's unique name or collection code are not found in the database, the endpoint does not return any record.

Validate an attribute

Users that are already aware of the format of the biological source attributes, and have information about the institution and collection codes, may use the API endpoint *Validate Attribute* to validate the attribute string. The user needs to provide the attribute string in the format detailed in [Table 1](#), according to the attribute type. The attribute type ('qualifier_type': specimen voucher, culture collection, or bio material) may also be specified to narrow the search, but if none is provided the API will search within all attribute values ([Table 8](#)). The API performs the search for the exact match, but if none is found, a search for partial matches for the provided string will be performed. The response includes the type of match (match level exact or partial), a recommendation for the qualifier value (match) that may correspond to the input or include corrections to the unique values of the institution and collections, and the metadata of the referred institution and collections ([Figure 8](#)). If the match isn't exact and there is more than one possible match to the attribute string input by the user, the response will include all possible matches and associated metadata.

Construct the attribute

Users may use the API endpoint *Construct the Attribute* to help them obtain the correct attribute string for referring to the biological source of the voucher linked with the sequence data. In this endpoint the user needs to provide separately the expected values for the institution, collection, and the ID of the specimen, culture, or material, depending on the attribute type ([Table 9](#)). The type of attribute ('qualifier_type': specimen voucher, culture collection, or bio material) may also be specified to narrow the search, but if none is provided the API will search within all attribute values. As in the

Table 8. Parameters required for the API endpoint *Validate attribute*. Requirements and an example are also provided. API, Application Programming Interface.

Parameters	Example	Type	Requirement	Notes
value	MSNT:FAZC:123456	String	Mandatory	Attribute string to be validated in the format described in Table 1
qualifier_type	specimen_voucher, bio_material, culture_collection	String	Optional	Filters results for a specific attribute

Table 9. Parameters required for the API endpoint *Construct the Attribute*. Requirements and an example are also provided. API, Application Programming Interface.

Parameters	Example	Type	Requirement	Notes
ivalue	HSUV	String	Mandatory	Institution's unique name
cvalue	Bird	String	Optional	Collection Code
id	123456	String	Mandatory	Identifier value
qualifier_type	specimen_voucher, bio_material, culture_collection	String	Optional	Filters results for a specific attribute

validate function, the API also searches for partial matches. The response includes the type of match (match level exact or partial), the constructed attribute string from the values input by the user (input value), a recommendation for the attribute string (match) that may correspond to the input or include corrections to the unique values of the institution and collections, and the metadata of the referred institution and collections. In [Figure 7](#) (cURL request for /construct API endpoint) we can see a case where the match is only partial, as there is more than one option for the institution code provided by the user. In these situations, the response will include all possible matches and associated metadata.

Future steps

Further developments of the ENA Source Attribute Helper API are planned.

Regarding the retrieval of the data, an automated flow for getting the updated files from the NCBI servers regularly is planned for implementation.

The development of a Graphical User Interface (GUI) is also planned for implementation, likely embedded in, or accessible from, one or more of ENA's existing submission tools. This will allow more intuitive searches for Institutions and/or Collections metadata and the validation/construction of the qualifier values to be more accessible to inexperienced users. This UI will connect to the API to support features like:

- Dynamic auto completion of user input
- Visual indicators for attribute matches
- Easy copying of constructed/validated attributes
- Metadata browsing

Conclusions

Considering the increasing rates of generation and submission of sequence data to public repositories it becomes increasingly important to assure the greatest accuracy and precision of associated metadata. Hence, we have developed and deployed a tool that will considerably help users to provide accurate metadata for reference to the biological source of sequence data. We have described the ENA Source Attribute Helper API design and implementation and discussed its main usages. We expect this tool to promote and support the submission of better structured and more richly described data that will provide a stronger foundation to strengthen the value of natural history collections, taxonomic expertise, and biodiversity knowledge.

For biodiversity research, the wider availability of correctly structured biological source attributes in sequence data will, for instance, improve the linkage with distribution data in the Global Biodiversity Information Facility (GBIF; GBIF 2022). GBIF holds a data-clustering feature that identifies records that are potentially related by matching similar metadata entries (GBIF 2020). In the case of the INSDC Sequences dataset in GBIF the fields used for matching are the biological source attributes in the triple Darwin Core format (Grosjean & Robertson 2021). Therefore, we expect that the number of sequence records linked to specimens in natural history collections and to their distribution data will increase with the usage of the ENA Source Attribute Helper API. Monitoring these links will help us to measure the impact of the usage of this tool.

Overall, we expect the enrichment of the provenance metadata of sequences in molecular biology repositories to contribute to boost our understanding of, and effectiveness of response to global challenges such as biodiversity loss, ecosystem change and food security.

Data availability

Underlying data

The data used in this API are available at <https://www.ncbi.nlm.nih.gov/biocollections>. The API retrieves data from the institutions, collections and unique institutions codes files that are available for public access at <https://ftp.ncbi.nih.gov/pub/taxonomy/biocollections/>.

Software availability

Software available from: <https://www.ebi.ac.uk/ena/sah/api/>

Source code available from: <https://github.com/enasequence/ena-source-annotation-helper>

Archived source code at time of publication: <https://doi.org/10.5281/zenodo.7063227> (Jayathilaka & Gupta 2022)

License: [Apache License 2.0 license](#)

Acknowledgements

We would like to thank Conrad Schoch, Shobba Sharma and the NCBI taxonomy team for providing information on the NCBI biocollections database and its ftp access. We would also like to acknowledge Mathias Dillen, Marcus Ernst, Quentin Groom, Anton Güntsch and Tim Robertson for their inputs to an earlier version of the manuscript and to the BiCIKL Consortium for their support.

References

- Arita M, Karsch-Mizrachi I, Cochrane G: **The international nucleotide sequence database collaboration**. *Nucleic Acids Res.* 2021; **49**: D121–D124.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Cummins C, Ahamed A, Aslam R, *et al.*: **The European Nucleotide Archive in 2021**. *Nucleic Acids Res.* 2022; **50**: D106–D110.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Fukuda A, Kodama Y, Mashima J, *et al.*: **DDBJ update: streamlining submission and access of human data**. *Nucleic Acids Res.* 2021; **49**: D71–D75.
[PubMed Abstract](#) | [Publisher Full Text](#)
- GBIF: **The Global Biodiversity Information Facility: What is GBIF?** 2022. (accessed 29 June 2022).
[Reference Source](#)
- GBIF: **New data-clustering feature aims to improve data quality and reveal cross-dataset connections**. *News.* 2020. 28 July 2020.
[Reference Source](#)
- Grosjean M, Robertson T: *Identifying potentially related records - How does the GBIF data-clustering feature work?* GBIF data blog; 2021.
[Reference Source](#)
- INSDC: **The DDBJ/ENA/GenBank Feature Table Definition. Version 11.1 October 2021**. 2021.
[Reference Source](#)
- Jayathilaka S, Gupta V: **ENA Source Attribute Helper (v1.0.2)**. [Software]. **Zenodo**. 2022.
[Publisher Full Text](#)
- Penev L, Koureas D, Groom Q, *et al.*: **Biodiversity Community Integrated Knowledge Library (BiCIKL)**. *Res. Ideas Outcomes.* 2022; **8**: e811360.
[Publisher Full Text](#)
- Sayers EW, Cavanaugh M, Clark K, *et al.*: **GenBank**. *Nucleic Acids Res.* 2021; **49**: D92–D96.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Sharma S, Ciuffo S, Starchenko E, *et al.*: **The NCBI Biocollections Database**. *Database.* 2018; **2018**: bay006.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Waterhouse RM, Adam-Blondon AF, Agosti D, *et al.*: **Recommendations for connecting molecular sequence and biodiversity research infrastructures through ELIXIR [version 1; peer review: awaiting peer review]**. *F1000Res.* 2021; **10**(ELIXIR): 1238.
[Publisher Full Text](#)
- Wieczorek J, Bloom D, Guralnick R, *et al.*: **Darwin Core: An Evolving Community-Developed Biodiversity Data Standard**. *PLoS One.* 2012; **7**(1): e29715.
[PubMed Abstract](#) | [Publisher Full Text](#)

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research