Search API FAQ

Search API Frequently Asked Questions (FAQ)

Contents:

Q 1: Where is the documentation for the Search API?

See Search Service API documentation.

Q 2: What is the baseURL for the Search API?

The baseURL is http://nsdl.org/dds-search

Q 3: Can I filter the search results by Education Level, Audience, Resource Type, Access Rights, etc.?

Yes. Education Level, Resource Type, Access Rights, Language, Medium, and MimeType are fixed controlled vocabularies that are normalized in the NSDL DC metadata and search index. Subject also has a controlled vocabulary but the metadata is not filtered so there are other subject vocabularies and keyword that appear in that field.

To filter a search by a given field/term, add a Boolean clause to the query supplied in the 'q' argument, for example (*user's typed keywords*) AND /key //nsdl_dc/educationLevel:"Middle School" AND xmlFormat:nsdl_dc

For example, the following request performs a search for '*video*' filtered to *Middle School* education level only:
http://nsdl.org/dds-search?verb=Search&q=(video) AND /key//nsdl_dc/educationLevel:"Middle School" AND xmlFormat:nsdl_dc&s=0&n=10

This request will search for *Movie/Animation* in the type field (instead of searching the record for the word 'video') and *Middle School* education level:
http://nsdl.org/dds-search?verb=Search&q=/key//nsdl_dc/type:%28Movie/Animation%29+AND+/key//nsdl_dc/educationLevel:%22Middle+School%22+AND+xmlFormat:nsdl_dc&s=0&n=10

Fields that start with /key//nsdl_dc/ correspond to metadata elements in the NSDL DC metadata framework, which represents the educational resources in the library. See this NSDL DC metadata guide for details.

Use the ListTerms API request to fetch the list of available terms for each field (to get a list of fields, use ListFields).

For example, these requests list the terms/vocabs for Education Level and Type, respectively: http://nsdl.org/dds-search?verb=ListTerms&field=/key//nsdl_dc/educationLevel http://nsdl.org/dds-search?verb=ListTerms&field=/key//nsdl_dc/type

Q 4: Which XML format should I use?

The repository contains multiple types of records that are expressed in different XML formats. The canonical XML format for educational resources is NSDL DC (`nsdl_dc`), which should be used for most search applications. Other formats that appear in the repository are `comm_anno`, which contain annotation data (comments, tips, reviews, etc.), `comm_para`, which contains usage summary data (favorited 12 times, etc.), Learning Application Ready `lar` metadata, which is richer format that describes educational resources, and `dlese_collect`, which contains metadata about the collections in the repository. Note that for each `lar` metadata record in the repository, there is a corresponding `nsdl_dc` record.

To filter your search to only the canonical NSDL DC format, add a Boolean query clause to your search query to require `xmlFormat:nsdl_dc`. For example: `(user keywords) AND (xmlFormat:nsdl_dc)`. This will filter the search to only those resources that have one or more `nsdl_dc` record for them.

Q 5: How does NSDL.org use the API to implement its search pages?

The search and browse pages at NSDL.org are implemented using the Search API. The UI provides keyword search as well as options to search and filter by Grade Level, Resource Type, Subject, and Pathway. For example, see this search for 'ocean' with 'Grade Level: Elementary School' selected.

To see the full API request that is used for a given search in the NSDL.org UI, first perform a search like you normally would and then add the http param &showquery=1 to the url in your browser, for example: http://nsdl.org/search/index.php?q=ocean&submitButton=Go&n=10&educationLevel%5B%5D=Elementary+School&showquery=1

The q argument in the API request contains the search query, for example from the above: (((ocean) OR stems:(ocean) OR title:(ocean) OR titlestems:(ocean) OR description:(ocean) OR descriptionstems:(ocean))) AND (/key//nsdl_dc/educationLevel "Elementary School")) AND xmlFormat:nsdl_dc

Here the term 'ocean' is expanded to search across the default, stems, title, titlestems, description and descriptionstems fields, which is done to boost the rank ordering for matches in the title, titlestems, description and descriptionstems fields of the record while also including matches in the default and stems fields.

The next clause in the query, AND (/key//nsdl_dc/educationLevel "Elementary School"), filters the search to education/grade level "Elementary School". The final clause, AND xmlFormat:nsdl_dc, filters the search to NSDL DC records only, which are the educational resources in the library.

Refer to the API documentation for information about what the other request parameters are used for. NSDL.org uses some advanced features from DDS, which may not be needed for many applications. Play around with this UI and compare the request that is being sent to see what is going on.

Q 6: How can I apply my own query expansion and boosting in the Search request?

The Search request allows you to construct Lucene queries that operate over any field in the search index, supplied in the q argument of the request. A text query without any explicit field specifier is applied to the default field, which contains the full text of the resource metadata and the crawled content of the front page of the resource itself. Applying a user's search terms to the default field produces fairly good results but it is often better to apply query expansion techniques such as stemming to increase the scope of the search and to perform boosting to augment the ordering of the results to bring more relevant resources to the top of the hit list.

The index contains a number of search fields that are useful for query expansion and boosting. These include but are not limited to:

- stems - Contains the same content as the default field but in stemmed form
- title - Contains the title of the resource
- titlestems - Contains the title of the resource in stemmed form
- description - Contains the description of the resource
- descriptionstems - Contains the description of the resource in stemmed form

To apply query expansion, construct a Boolean query using the Lucene syntax to expand the user's query across the desired fields, and then supply this in the q argument in the Search request. For example, if the user has typed a search for the word *ocean*, the following progression illustrates how the expanded query might be constructed:

- (ocean) - The raw user search terms without any query expansion applied. This will match results that have the exact word *ocean* in the default field.

- (ocean) OR stems:(ocean) - Expands the query to also match word stems. This will match results that have the exact word *ocean* in them as well as morphologically similar words including *oceans* and *oceanic.*
- (ocean) OR stems:(ocean) OR title:(ocean) - Adds boosting to provide more weight to those results that contain the exact word *ocean* in the title. This will match the same results as the previous example but provide them in a different rank order.

- (ocean) OR stems:(ocean) OR title:(ocean) OR titlestems:(ocean) - Adds additional boosting to provide weight to titles that contain morphologically similar words to *ocean* in the title and still more weight for exact matches for *ocean* in the title. This will match the same results as the previous example but provide them in a different rank order.

- (ocean) OR stems:(ocean) OR title:(ocean) OR titlestems:(ocean) OR description:(ocean) OR descriptionstems:(ocean) - Adds weight for matches in the description and stemmed version of description as well, which will match the same results as the previous example but provide them in a different rank order.

Note that for the stemmed fields it is not necessary to stem the words supplied in the query - just supply the user's text exactly as it was typed and the Search API will automatically apply word stemming in the appropriate fields for you. Because the colon is a reserved character in the Lucene syntax, these and other reserved characters should be filtered out prior to issuing the query, unless you want to allow the user to perform Lucene queries themselves.

One more Boolean clause that is often desirable is to restrict the search to educational resources only by adding a clause xmlFormat:nsdl_dc to the query:

- ((ocean) OR stems:(ocean) OR title:(ocean) OR titlestems:(ocean) OR description:(ocean) OR descriptionstems:(ocean)) AND xmlFormat:nsdl_dc - Restricts the search to educational resources in the NSDL DC format only.

This final example is the query expansion technique used for user's searches at nsdl.org.

Q 7: Is there a query to show me my collection's resources that I do or don't have paradata records for?

The following query will show you your collection's resources that are referenced by paradata collections from a certain collection (i.e. your own paradata collection). This particular example uses the SMILE resource and paradata collection keys (ncs-NSDL-COLLECTION-000-003-112-056 and ncs-NSDL-COLLECTION-000-003-112-075 respectively). Replace these with your own collection keys. If you are not sure what the collection key is, use the search request http://nsdl.org/dds-search?verb=ListCollections to find the <searchKey> for your collection.

http://nsdl.org/dds-search?verb=Search&q=(isRelatedToByCollectionKey:"ncs-NSDL-COLLECTION-000-003-112-075")&ky=ncs-NSDL-COLLECTION-000-003-112-056&s=0&n=100

This query, on the other hand, will show the resources from your collection that are not associated with paradata from your paradata collection (again, replace collection keys with your own).

http://nsdl.org/dds-search?verb=Search&q=(*:* NOT isRelatedToByCollectionKey:"ncs-NSDL-COLLECTION-000-003-112-075")&ky=ncs-NSDL-COLLECTION-000-003-112-056&s=0&n=100 (http://nsdl.org/dds-search?verb=Search&q=(*:*%20NOT%20isRelatedToByCollectionKey:%22ncs-NSDL-COLLECTION-000-003-112-075%22)&ky=ncs-NSDL-COLLECTION-000-003-112-056&s=0&n=100)

Q 8: How do I search for a specific url across all collections in the NSDL?

http://nsdl.org/dds-search?verb=Search&q=url:%22http://eosweb.larc.nasa.gov/%22&s=0&n=10&relation=alsoCatalogedBy&response=allCollectionsMetadata
Replace http://eosweb.larc.nasa.gov/ in the above query with the url you are searching for.

Q 9: How do I find all records in the NSDL that have standards aligned to them?

This query will give you all nsdl_dc records with ASN standards aligned (adjust n=100 for # of results shown):
http://nsdl.org/dds-search?verb=Search&q=/key//nsdl_dc/conformsTo:(http*asn.jesandco.org/%20OR%20http*purl.org)%20OR%20/relation.isAnnotatedBy//key//comm_anno/ASNstandard:(http*asn.jesandco.org/%20OR%20http*purl.org)&s=0&n=100 (http://nsdl.org/dds-search?verb=Search&q=/key//nsdl_dc/conformsTo:(http*asn.jesandco.org/*%20OR%20http*purl.org*)%20OR%20/relation.isAnnotatedBy//key//comm_anno/ASNstandard:(http*asn.jesandco.org/*%20OR%20http*purl.org*)&s=0&n=100)

This query will give you all comm_anno records with ASN standards aligned (adjust n=100 for # of results shown):
http://nsdl.org/dds-search?verb=Search&q=/key//comm_anno/ASNstandard:(http*asn.jesandco.org/%20OR%20http*purl.org)&s=0&n=100 (http://nsdl.org/dds-search?verb=Search&q=/key//comm_anno/ASNstandard:(http*asn.jesandco.org/*%20OR%20http*purl.org*)&s=0&n=100)

This query will give you all nsdl_dc records with Strand Map Service ID's associated with them:
http://nsdl.org/dds-search?verb=Search&q=/text//nsdl_dc/conformsTo:(SMS%20BMK*)&s=0&n=100

Q 10: Can I perform faceted searches?

Yes. See Faceted Search for details.

Q 11: Are search results deduped?

Yes. Deduping is applied to matching `nsdl_dc` results in the Search API and is designed to remove duplicate returns from being displayed in the search results at NSDL.org and other portals/applications built on the Search API. By default the API returns the single `nsdl_dc` metadata record that best matches the search criteria. For example, suppose a given resource URL has multiple `nsdl_dc` metadata entries that describe it, one from collection A, another from collection B, and a third from collection C. When a search is performed that matches the resource, the results will return the single nsdl_dc metadata record that best matches the search criteria, and it may return different metadata for different searches depending on the terms and field classifiers indicated in the search. If the search indicates collection A as it's sole search criteria for example, the API will return the metadata from partner A, likewise if partner B is indicated, collection B's metadata will be returned. Best match is calculated using standard IR algorithms applied to all search criteria indicated including search terms, fielded classifiers (Grade Range, Subject, etc.), and collection, compared against the metadata features (terms and classifiers) that appear in the records.

By default only the best matching `nsdl_dc` metadata record is returned. To have all available `nsdl_dc` metadata returned, however, include the argument `&relation=alsoCatalogedBy` in the `Search` request. If more than one `nsdl_dc` metadata record is available for given `Search` result, they will appear in the `<relations>` portion of the XML response.