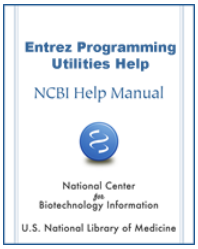
U.S. National Library of Medicine
National Center for Biotechnology Information

NLM Citation: Sayers E. A General Introduction to the E-utilities. In: Entrez Programming Utilities Help [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2010-.
Bookshelf URL: https://www.ncbi.nlm.nih.gov/books/



A General Introduction to the E-utilities

Eric Sayers, PhD[1]

Introduction

The Entrez Programming Utilities (E-utilities) are a set of nine server-side programs that provide a stable interface into the Entrez query and database system at the National Center for Biotechnology Information (NCBI). The E-utilities use a fixed URL syntax that translates a standard set of input parameters into the values necessary for various NCBI software components to search for and retrieve the requested data. The E-utilities are therefore the structured interface to the Entrez system, which currently includes 38 databases covering a variety of biomedical data, including nucleotide and protein sequences, gene records, three-dimensional molecular structures, and the biomedical literature.

To access these data, a piece of software first posts an E-utility URL to NCBI, then retrieves the results of this posting, after which it processes the data as required. The software can thus use any computer language that can send a URL to the E-utilities server and interpret the XML response; examples of such languages are Perl, Python, Java, and C++. Combining E-utilities components to form customized data pipelines within these applications is a powerful approach to data manipulation.

This chapter first describes the general function and use of the eight E-utilities, followed by basic usage guidelines and requirements, and concludes with a discussion of how the E-utilities function within the Entrez system.

Usage Guidelines and Requirements

Use the E-utility URL

All E-utility requests should be made to URLs beginning with the following string:

https://eutils.ncbi.nlm.nih.gov/entrez/eutils/

These URLs direct requests to servers that are used only by the E-utilities and that are optimized to give users the best performance.

Frequency, Timing and Registration of E-utility URL Requests

In order not to overload the E-utility servers, NCBI recommends that users post no more than three URL requests per second and limit large jobs to either weekends or between 9:00 PM and 5:00 AM Eastern time

Author Affiliation: 1 NCBI; Email: sayers@ncbi.nlm.nih.gov.

Corresponding author.

during weekdays. Failure to comply with this policy may result in an IP address being blocked from accessing NCBI. If NCBI blocks an IP address, service will not be restored unless the developers of the software accessing the E-utilities register values of the **tool** and **email** parameters with NCBI. The value of **tool** should be a string with no internal spaces that uniquely identifies the software producing the request. The value of **email** should be a complete and valid e-mail address of the software developer and not that of a third-party end user. The value of **email** will be used only to contact developers if NCBI observes requests that violate our policies, and we will attempt such contact prior to blocking access. In addition, developers may request that the value of **email** be added to the E-utility mailing list that provides announcements of software updates, known bugs and other policy changes affecting the E-utilities. To register **tool** and **email** values, simply send an e-mail to eutilities@ncbi.nlm.nih.gov including the desired values along with the name of either a developer or the organization creating the software. Once NCBI establishes communication with a developer, receives values for **tool** and **email** and validates the e-mail address in **email**, the block will be lifted. Once **tool** and **email** values are registered, all subsequent E-utility requests from that software package should contain both values. Please be aware that merely providing values for **tool** and **email** in requests is not sufficient to comply with this policy; these values must be registered with NCBI. Requests from any IP that lack registered values for **tool** and **email** and that violate the above usage policies may be blocked. Software developers may register values of **tool** and **email** at any time, and are encouraged to do so.

Coming in December 2018: API Keys

On December 1, 2018, NCBI will begin enforcing the use of API keys that will offer enhanced levels of supported access to the E-utilities. After that date, any site (IP address) posting more than 3 requests per second to the E-utilities without an API key will receive an error message. By including an API key, a site can post up to 10 requests per second by default. Higher rates are available by request (eutilities@ncbi.nlm.nih.gov). Users can obtain an API key now from the Settings page of their NCBI account (to create an account, visit http://www.ncbi.nlm.nih.gov/account/). After creating the key, users should include it in each E-utility request by assigning it to the new *api_key* parameter.

Example request including an API key:
```
esummary.fcgi?db=pubmed&id=123456&api_key=ABCDE12345
```

Example error message if rates are exceeded:
```
{"error":"API rate limit exceeded","count":"11"}
```

Only one API key is allowed per NCBI account; however, a user may request a new key at any time. Such a request will invalidate any existing API key associated with that NCBI account.

We encourage regular E-utility users to obtain an API key as soon as possible and begin the process of incorporating it into code. We also encourage users to monitor their request rates to determine if they will require rates higher than 10 per second. As stated above, we can potentially have higher rates negotiated prior to the beginning of enforcement on December 1, 2018.

Minimizing the Number of Requests

If a task requires searching for and/or downloading a large number of records, it is much more efficient to use the Entrez History to upload and/or retrieve these records in batches rather than using separate requests for each record. Please refer to Application 3 in Chapter 3 for an example. Many thousands of IDs can be uploaded using a single EPost request, and several hundred records can be downloaded using one EFetch request.

Disclaimer and Copyright Issues

If you use the E-utilities within software, NCBI's Disclaimer and Copyright notice (https://www.ncbi.nlm.nih.gov/About/disclaimer.html) must be evident to users of your product. Please note that

abstracts in PubMed may incorporate material that may be protected by U.S. and foreign copyright laws. All persons reproducing, redistributing, or making commercial use of this information are expected to adhere to the terms and conditions asserted by the copyright holder. Transmission or reproduction of protected items beyond that allowed by fair use (PDF) as defined in the copyright laws requires the written permission of the copyright owners. NLM provides no legal advice concerning distribution of copyrighted materials. Please consult your legal counsel. If you wish to do a large data mining project on PubMed data, you can enter into a licensing agreement and lease the data for free from NLM. For more information on this please see http://www.nlm.nih.gov/databases/leased.html.

Handling Special Characters Within URLs

When constructing URLs for the E-utilities, please use lowercase characters for all parameters except &WebEnv. There is no required order for the URL parameters in an E-utility URL, and null values or inappropriate parameters are generally ignored. Avoid placing spaces in the URLs, particularly in queries. If a space is required, use a plus sign (+) instead of a space:

```
Incorrect: &id=352, 25125, 234
Correct:   &id=352,25125,234

Incorrect: &term=biomol mrna[properties] AND mouse[organism]
Correct:   &term=biomol+mrna[properties]+AND+mouse[organism]
```

Other special characters, such as quotation marks (“) or the # symbol used in referring to a query key on the History server, should be represented by their URL encodings (%22 for “; %23 for #).

```
Incorrect: &term=#2+AND+"gene in genomic"[properties]
Correct:   &term=%232+AND+%22gene+in+genomic%22[properties]
```

The Nine E-utilities in Brief

EInfo (database statistics)

eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi

Provides the number of records indexed in each field of a given database, the date of the last update of the database, and the available links from the database to other Entrez databases.

ESearch (text searches)

eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi

Responds to a text query with the list of matching UIDs in a given database (for later use in ESummary, EFetch or ELink), along with the term translations of the query.

EPost (UID uploads)

eutils.ncbi.nlm.nih.gov/entrez/eutils/epost.fcgi

Accepts a list of UIDs from a given database, stores the set on the History Server, and responds with a query key and web environment for the uploaded dataset.

ESummary (document summary downloads)

eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi

Responds to a list of UIDs from a given database with the corresponding document summaries.

EFetch (data record downloads)

eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi

Responds to a list of UIDs in a given database with the corresponding data records in a specified format.

ELink (Entrez links)

eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi

Responds to a list of UIDs in a given database with either a list of related UIDs (and relevancy scores) in the same database or a list of linked UIDs in another Entrez database; checks for the existence of a specified link from a list of one or more UIDs; creates a hyperlink to the primary LinkOut provider for a specific UID and database, or lists LinkOut URLs and attributes for multiple UIDs.

EGQuery (global query)

eutils.ncbi.nlm.nih.gov/entrez/eutils/egquery.fcgi

Responds to a text query with the number of records matching the query in each Entrez database.

ESpell (spelling suggestions)

eutils.ncbi.nlm.nih.gov/entrez/eutils/espell.fcgi

Retrieves spelling suggestions for a text query in a given database.

ECitMatch (batch citation searching in PubMed)

eutils.ncbi.nlm.nih.gov/entrez/eutils/ecitmatch.cgi

Retrieves PubMed IDs (PMIDs) corresponding to a set of input citation strings.

Understanding the E-utilities Within Entrez

The E-utilities Access Entrez Databases

The E-utilities access the core search and retrieval engine of the Entrez system and, therefore, are only capable of retrieving data that are already in Entrez. Although the majority of data at NCBI are in Entrez, there are several datasets that exist outside of the Entrez system. Before beginning a project with the E-utilities, check that the desired data can be found within an Entrez database.

The Entrez System Identifies Database Records Using UIDs

Each Entrez database refers to the data records within it by an integer ID called a UID (unique identifier). Examples of UIDs are GI numbers for Nucleotide and Protein, PMIDs for PubMed, or MMDB-IDs for Structure. The E-utilities use UIDs for both data input and output, and thus it is often critical, especially for advanced data pipelines, to know how to find the UIDs associated with the desired data before beginning a project with the E-utilities.

See Table 1 for a complete list of UIDs in Entrez.

Table 1 – Entrez Unique Identifiers (UIDs) for selected databases

Entrez Database	UID common name	E-utility Database Name
BioProject	BioProject ID	bioproject

Table 1 continued from previous page.

Entrez Database	UID common name	E-utility Database Name
BioSample	BioSample ID	biosample
Biosystems	BSID	biosystems
Books	Book ID	books
Conserved Domains	PSSM-ID	cdd
dbGaP	dbGaP ID	gap
dbVar	dbVar ID	dbvar
Epigenomics	Epigenomics ID	epigenomics
EST	GI number	nucest
Gene	Gene ID	gene
Genome	Genome ID	genome
GEO Datasets	GDS ID	gds
GEO Profiles	GEO ID	geoprofiles
GSS	GI number	nucgss
HomoloGene	HomoloGene ID	homologene
MeSH	MeSH ID	mesh
NCBI C++ Toolkit	Toolkit ID	toolkit
NCBI Web Site	Web Site ID	ncbisearch
NLM Catalog	NLM Catalog ID	nlmcatalog
Nucleotide	GI number	nuccore
OMIA	OMIA ID	omia
PopSet	PopSet ID	popset
Probe	Probe ID	probe
Protein	GI number	protein
Protein Clusters	Protein Cluster ID	proteinclusters
PubChem BioAssay	AID	pcassay
PubChem Compound	CID	pccompound
PubChem Substance	SID	pcsubstance
PubMed	PMID	pubmed
PubMed Central	PMCID	pmc
SNP	rs number	snp
SRA	SRA ID	sra
Structure	MMDB-ID	structure
Taxonomy	TaxID	taxonomy
UniGene	UniGene Cluster ID	unigene
UniSTS	STS ID	unists

Accessing Sequence Records Using Accession.Version Identifiers

NCBI now uses the accession.version identifier rather that the GI number (UID) as the primary identifier for nucleotide and protein sequence records (records in the nuccore, nucest, nucgss, popset, and protein databases). Even so, the E-utilities continue to provide access to these records using either GI numbers or accession.version identifiers. Those E-utilities that accept UIDs as input will also accept accession.version identifiers (for the sequence databases listed above). Those E-utilities that output UIDs can output accession.version identifiers instead by setting the &idtype parameter to "acc". Finally, EFetch can retrieve *any* sequence record by its accession.version identifier, including sequences that do not have GI numbers. Please see Chapter 4 for more details about how each E-utility handles accession.version identifers.

The Entrez Core Engine: EGQuery, ESearch, and ESummary

The core of Entrez is an engine that performs two basic tasks for any Entrez database: 1) assemble a list of UIDs that match a text query, and 2) retrieve a brief summary record called a Document Summary (DocSum) for each UID. These two basic tasks of the Entrez engine are performed by ESearch and ESummary. ESearch returns a list of UIDs that match a text query in a given Entrez database, and ESummary returns DocSums that match a list of input UIDs. A text search in web Entrez is equivalent to ESearch-ESummary. EGQuery is a global version of ESearch that searches all Entrez databases simultaneously. Because these three E-utilities perform the two core Entrez functions, they function for all Entrez databases.

```
egquery.fcgi?term=query
esearch.fcgi?db=database&term=query
esummary.fcgi?db=database&id=uid1,uid2,uid3,...
```

Syntax and Initial Parsing of Entrez Queries

Text search strings entered into the Entrez system are converted into Entrez queries with the following format:

term1[field1] **Op** term2[field2] **Op** term3[field3] **Op** ...

where the terms are search terms, each limited to a particular Entrez field in square brackets, combined using one of three Boolean operators: Op = AND, OR, or NOT. These Boolean operators must be typed in all capital letters.

Example: human[organism] AND topoisomerase[protein name]

Entrez initially splits the query into a series of items that were originally separated by spaces in the query; therefore it is critical that spaces separate each term and Boolean operator. If the query consists *only* of a list of UID numbers (unique identifiers) or accession numbers, the Entrez system simply returns the corresponding records and no further parsing is performed. If the query contains any Boolean operators (AND, OR, or NOT), the query is split into the terms separated by these operators, and then each term is parsed independently. The results of these searches are then combined according to the Boolean operators.

A full account of how to search Entrez can be found in the Entrez Help Document. Additional information is available from Entrez Help.

Entrez Databases: EInfo, EFetch, and ELink

The NCBI Entrez system currently contains 38 databases. EInfo provides detailed information about each database, including lists of the indexing fields in the database and the available links to other Entrez databases.

```
einfo.fcgi?db=database
```

Each Entrez database includes two primary enhancements to the raw data records: 1) software for producing a variety of display formats appropriate to the given database, and 2) links to records in other Entrez databases manifested as lists of associated UIDs. The display format function is performed by EFetch, which generates formatted output for a list of input UIDs. For example, EFetch can produce abstracts from Entrez PubMed or FASTA format from Entrez Protein. EFetch does not yet support all Entrez databases; please see the EFetch documentation for details.

```
efetch.fcgi?db=database&id=uid1,uid2,uid3&rettype=report_type&retmode=
data_mode
```

The linking function is performed by ELink, which generates a list of UIDs in a specified Entrez database that are linked to a set of input UIDs in either the same or another database. For example, ELink can find Entrez SNP records linked to records in Entrez Nucleotide, or Entrez Domain records linked to records in Entrez Protein.

```
elink.fcgi?dbfrom=initial_databasedb=target_database&id=uid1,uid2,uid3
```

Using the Entrez History Server

A powerful feature of the Entrez system is that it can store retrieved sets of UIDs temporarily on the servers so that they can be subsequently combined or provided as input for other E-utility calls. The Entrez History server provides this service and is accessed on the Web using either the Preview/Index or History tabs on Entrez search pages. Each of the E-utilities can also use the History server, which assigns each set of UIDs an integer label called a query key (&query_key) and an encoded cookie string called a Web environment (&WebEnv). EPost allows any list of UIDs to be uploaded to the History Server and returns the query key and Web environment. ESearch can also post its output set of UIDs to the History Server, but only if the &usehistory parameter is set to "y". ELink also can post its output to the History server if &cmd is set to "neighbor_history". The resulting query key and Web environment from either EPost or ESearch can then be used in place of a UID list in ESummary, EFetch, and ELink.

In Entrez, a set of UIDs is represented on the History by three parameters:

```
&db = database; &query_key = query key; &WebEnv = web environment
```

Upload steps that generate a web environment and query key

```
esearch.fcgi?db=database&term=query&usehistory=y
```

```
epost.fcgi?db=database&id=uid1,uid2,uid3,...
```

```
elink.fcgi?dbfrom=source_db&db=destination_db&cmd=neighbor_history&id=
uid1,uid2,...
```

Download steps that use a web environment and query key

```
esummary.fcgi?db=database&WebEnv=webenv&query_key=key
```

```
efetch.fcgi?db=database&WebEnv=webenv&query_key=key&rettype=
report_type&retmode=data_mode
```

Link step that uses a web environment and query key

```
elink.fcgi?dbfrom=initial_databasedb=target_database&WebEnv=
webenv&query_key=key
```

Search step that uses a web environment and a query key in the &term parameter (preceded by #, encoded as %23)

```
esearch.fcgi?db=database&term=%23key+AND+query&WebEnv=webenv&usehistory=y
```

Generating Multiple Data Sets on the History Server

Each web environment on the History Server can be associated with any number of query keys. This allows different data sets to be combined with the Boolean operators AND, OR, and NOT, or with another Entrez query. It is important to remember that for two data sets (query keys) to be combined, they must be associated with the same web environment. By default, successive E-utility calls produce query keys that are *not* associated with the same web environment, and so to overcome this, each E-utility call after the initial call must set the &WebEnv parameter to the value of the pre-existing web environment.

Default behavior: These two URLs…

```
URL 1: epost.fcgi?db=database&id=uid1,uid2,uid3
URL 2: esearch.fcgi?db=database&term=query&usehistory=y
```

will produce two History sets associated with different web environments:

URL	WebEnv	query_key	UIDs
1	web1	1	uid1,uid2,uid3
2	web2	1	uids matching query

Desired behavior: These two URLs…

```
URL 1: epost.fcgi?db=database&id=uid1,uid2,uid3
(extract web1 from the output of URL 1)
URL 2: esearch.fcgi?db=database&term=query&usehistory=y&WebEnv=web1
```

will produce two sets associated with the same (new) web environment:

URL	WebEnv	query_key	UIDs
1	web2	1	uid1,uid2,uid3
2	web2	2	uids matching query

Combining E-utility Calls to Create Entrez Applications

The E-utilities are useful when used by themselves in single URLs; however, their full potential is realized when successive E-utility URLs are combined to create a data pipeline. When used within such pipelines, the Entrez History server simplifies complex retrieval tasks by allowing easy data transfer between successive E-utility calls. Listed below are several examples of pipelines produced by combining E-utilities, with the arrows representing the passing of db, WebEnv and query_key values from one E-utility to another. These and related pipelines are discussed in detail in Chapter 3.

Basic Pipelines

Retrieving data records matching an Entrez query

ESearch → ESummary

ESearch → EFetch

Retrieving data records matching a list of UIDs

EPost → ESummary

EPost → EFetch

Finding UIDs linked to a set of records

ESearch → ELink

EPost → ELink

Limiting a set of records with an Entrez query

EPost → ESearch

ELink → ESearch

Advanced Pipelines

Retrieving data records in database B linked to records in database A matching an Entrez query

ESearch → ELink → ESummary

ESearch → ELink → EFetch

Retrieving data records from a subset of an ID list defined by an Entrez query

EPost → ESearch → ESummary

EPost → ESearch → EFetch

Retrieving a set of data records, defined by an Entrez query, in database B from a larger set of records linked to a list of UIDs in database A

EPost → ELink → ESearch → ESummary

EPost → ELink → ESearch → EFetch

Demonstration Programs

Please see Chapter 1 for sample Perl scripts.

For More Information

Please see Chapter 1 for getting additional information about the E-utilities.