
eutils Documentation

Release 0.6.0

Reece Hart

Dec 18, 2019

Contents

1	Contents	1
1.1	Introduction	1
1.2	Using eutils	1
1.3	Modules	4
1.4	Contributing	7
1.5	ChangeLog	9
1.6	License	13
2	Indices and tables	19
	Python Module Index	21
	Index	23

1.1 Introduction

eutils is a Python package to simplify searching, fetching, and parsing records from NCBI using their [E-utilities](#) interface.

1.1.1 Features

- simple Pythonic interface for searching and fetching
- automatic query rate throttling per NCBI guidelines
- optional sqlite-based caching of compressed replies
- “façades” that facilitate access to essential attributes in XML replies

1.1.2 Important Notes

- **You are encouraged to [browse issues](#).** Please report any issues you find.
- **Use a pip package specification to ensure stay within minor releases for API stability.** For example, `eutils >=0.1,<0.2`.

1.2 Using eutils

1.2.1 Installation

The easiest way to install the eutils package is to use pre-build Python package from PyPI, like so:

```
$ pip install eutils
```

Consider using `virtualenvwrapper` or `pyenv` to setup virtual environments before installing `utils`.

Code that relies on `utils` should specify a version bracket to ensure that `utils` receives bug fixes but not updates that might break compatibility. In your package's `setup.py`:

```
setup(
    ...
    install_requires = [
        'utils>=0.4,<0.3',
    ],
    ...
)
```

Alternatively, you may install from source; please see *Installation for Development* for details.

1.2.2 Examples

Common setup

Instantiating an `utils` `utils.Client` is this easy:

```
>>> import utils

# Initialize a client. This client handles all caching and query
# throttling
>>> ec = utils.Client()
```

Fetching gene information

```
# search for tumor necrosis factor genes
# any valid NCBI query may be used
>>> esr = ec.esearch(db='gene',term='tumor necrosis factor')

# fetch one of those (gene id 7157 is human TNF)
>>> egs = ec.efetch(db='gene',id=7157)

# One may fetch multiple genes at a time. These are returned as an
# EntrezgeneSet. We'll grab the first (and only) child, which returns
# an instance of the Entrezgene class.
>>> eg = egs.entrezgenes[0]

# Easily access some basic information about the gene
>>> eg.hgnc, eg.maploc, eg.description, eg.type, eg.genus_species
('TP53', '17p13.1', 'tumor protein p53', 'protein-coding', 'Homo sapiens')

# get a list of genomic references
>>> [str(r) for r in eg.references]
['GeneCommentary(acv=NC_000017.11,type=genomic,heading=Reference GRCh38.p2 Primary_
↳Assembly,label=Chromosome 17 Reference GRCh38.p2 Primary Assembly)',
 'GeneCommentary(acv=NG_017013.2,type=genomic,heading=None,label=RefSeqGene)',
 'GeneCommentary(acv=NC_018928.2,type=genomic,heading=Alternate CHM1_1.1,
↳label=Chromosome 17 Alternate CHM1_1.1)']

# Get all products defined on GRCh38
>>> [p.acv for p in eg.references[0].products]
```

(continues on next page)

(continued from previous page)

```
[u'NM_001126112.2', u'NM_001276761.1', u'NM_000546.5',
u'NM_001276760.1', u'NM_001126113.2', u'NM_001276695.1',
u'NM_001126114.2', u'NM_001276696.1', u'NM_001126118.1',
u'NM_001126115.1', u'NM_001276697.1', u'NM_001126117.1',
u'NM_001276699.1', u'NM_001126116.1', u'NM_001276698.1']

# As a sample, grab the first product defined on this reference (order is arbitrary)
>>> mrna = eg.references[0].products[0]
>>> str(mrna)
'GeneCommentary(acv=NM_001126112.2,type=mRNA,heading=Reference,label=transcript_
↳variant 2) '

# mrna.genomic_coords provides access to the exon definitions on this
reference

>>> mrna.genomic_coords.gi, mrna.genomic_coords.strand
('568815581', -1)

>>> mrna.genomic_coords.intervals
[(7687376, 7687549), (7676520, 7676618), (7676381, 7676402),
(7675993, 7676271), (7675052, 7675235), (7674858, 7674970),
(7674180, 7674289), (7673700, 7673836), (7673534, 7673607),
(7670608, 7670714), (7668401, 7669689)]

# and the mrna has a product, the resulting protein:
>>> str(mrna.products[0])
'GeneCommentary(acv=NP_001119584.1,type=peptide,heading=Reference,label=isoform a) '
```

Fetch PubMed information

```
# search pubmed by author
>>> esr = c.esearch(db='pubmed', term='Nussbaum RL')

# fetch all of them
>>> paset = c.efetch(db='pubmed', id=esr.ids)

# paset represents PubmedArticleSet, a collection of
PubmedArticles. The major interface component is to iterate over
PubmedArticles with constructs like `for pa in paset: ...`. We
fetch the first PubmedArticle like this:
>>> pa = iter(paset).next()

PubmedArticle provides accessors to essential data:
>>> pa.title
'High incidence of functional ion-channel abnormalities in a
consecutive Long QT cohort with novel missense genetic variants of
unknown significance.'

>>> pa.authors
[u'Steffensen AB', u'Refaat MM', u'David JP', u'Mujezinovic A',
u'Calloe K', u'Wojciak J', u'Nussbaum RL', u'Scheinman MM',
u'Schmitt N']

>>> pa.jrnl, pa.volume, pa.issue, pa.year
('Sci Rep', '5', None, '2015')
```

(continues on next page)

(continued from previous page)

```
>>> pa.jrnl, pa.volume, pa.issue, pa.year, pa.pages
('Sci Rep', '5', None, '2015', '10009')

>>> pa.pmid, pa.doi, pa.pmc
('26066609', '10.1038/srep10009', '4464365')
```

1.3 Modules

1.3.1 Main Classes

class `eutils.Client` (*cache=False, api_key=None*)

Bases: `object`

class-based access to NCBI E-Utilities, returning Python classes with rich data accessors

Parameters

- **cache** (*str*) – passed to QueryService, which see for explanation
- **api_key** (*str*) – API key from NCBI

Raises `EutilsError` – if cache file couldn't be created

databases

list of databases available from eutils (per einfo query)

efetch (*db, id*)

query the efetch endpoint

einfo (*db=None*)

query the einfo endpoint

Parameters *db* – string (optional)

Return type EInfo or EInfoDB object

If *db* is `None`, the reply is a list of databases, which is returned in an EInfo object (which has a `databases()` method).

If *db* is not `None`, the reply is information about the specified database, which is returned in an EInfoDB object. (Version 2.0 data is automatically requested.)

esearch (*db, term*)

query the esearch endpoint

class `eutils.QueryService` (*email='biocommons-dev@googlegroups.com', cache=False, default_args={'retmax': 250, 'retmode': 'xml', 'usehistory': 'y'}, request_interval=None, tool=None, api_key=None*)

Bases: `object`

provides throttled and cached querying of NCBI E-utilities services

QueryService has three functions:

- construct URLs appropriate for eutils endpoints
- throttle queries per NCBI guidelines
- cache results in persistent cache (sqlite)

QueryService works with any valid query arguments, passed as dictionaries.

Implemented interfaces:

- `esearch`
- `efetch`
- `elink`
- `einfo`
- `esummary`

Implementing other query modes should be straightforward.

See also the NCBI's Entrez Programming Utilities Help: <http://www.ncbi.nlm.nih.gov/books/NBK25500/>

Parameters

- **email** (*str*) – email of user (for abuse reports)
- **cache** (*str*) – if True, cache at `~/.cache/eutils-db.sqlite`; if string, cache there; if False, don't cache
- **default_args** (*dict*) – dictionary of query args that should accompany all requests
- **request_interval** (*int or a callable returning an int*) – seconds between requests; default: auto-select based on API key
- **api_key** (*str*) – api key assigned by NCBI
- **tool** (*str*) – name of client

Return type `None`

Raises `OSError` – if sqlite file can't be opened

efetch (*args*)

execute a cached, throttled efetch query

Parameters **args** (*dict*) – dict of query items

Returns content of reply

Return type `str`

Raises `EutilsRequestError` – when NCBI replies, but the request failed (e.g., bogus database name)

einfo (*args=None*)

execute a NON-cached, throttled einfo query

`einfo.fcgi?db=<database>`

Input: Entrez database (&db) or None (returns info on all Entrez databases)

Output: XML containing database statistics

Example: Find database statistics for Entrez Protein.

`QueryService.einfo({"db": "protein"})`

Equivalent HTTP request:

<https://eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi?db=protein>

Parameters **args** (*dict*) – dict of query items (optional)

Returns content of reply

Return type `str`

Raises `EutilsRequestError` – when NCBI replies, but the request failed (e.g., bogus database name)

elink (*args*)

execute a cached, throttled elink query

Input: List of UIDs (&id); Source Entrez database (&dbfrom); Destination Entrez database (&db)

Output: XML containing linked UIDs from source and destination databases

Example: Find one set of Gene IDs linked to nuccore GIs 34577062 and 24475906

QueryService.elink({"dbfrom": "nuccore", "db": "gene", "id": "34577062,24475906"})

Equivalent HTTP request:

<https://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=nuccore&db=gene&id=34577062,24475906>

Parameters **args** (*dict*) – dict of query items containing at least the “db”, “dbfrom”, and “id” keys.

Returns content of reply

Return type `str`

Raises `EutilsRequestError` – when NCBI replies, but the request failed (e.g., bogus database name)

esearch (*args*)

execute a cached, throttled esearch query

Parameters **args** (*dict*) – dict of query items, containing at least “db” and “term” keys

Returns content of reply

Return type `str`

Raises `EutilsRequestError` – when NCBI replies, but the request failed (e.g., bogus database name)

esummary (*args*)

execute a cached, throttled esummary query

Input: List of UIDs (&id); Entrez database (&db)

Output: XML document summary for requested ID(s) [comma-separated]

Example:

QueryService.esummary({"db": "medgen", "id": 134 })

Equivalent HTTP request:

<https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=medgen&id=134>

Parameters **args** (*dict*) – dict of query items containing at least “db” and “id” keys.

Returns content of reply

Return type `str`

Raises *EutilsRequestError* – when NCBI replies, but the request failed (e.g., bogus database name)

1.3.2 Exceptions

class `utils.EutilsError`

Bases: `Exception`

Base class for all Eutils exceptions, and also used to raise general exception.

class `utils.EutilsNCBIError`

Bases: `utils._internal.exceptions.EutilsError`

Raised when NCBI returns data that appears to be incorrect or invalid.

class `utils.EutilsNotFoundError`

Bases: `utils._internal.exceptions.EutilsError`

Raised when the requested data is not available. (Used only by the *utils.sketchy.clientx* interface currently.)

class `utils.EutilsRequestError`

Bases: `utils._internal.exceptions.EutilsError`

Raised when NCBI responds with an error, such as when a non-existent database is specified.

1.3.3 Experimental

class `utils.sketchy.clientx.ClientX(cache=False, api_key=None)`

Bases: `utils._internal.client.Client`

warning This class is subject to rapid development and api changes.

A subclass of `utils.client.Client` that provides specific lookup functions.

This functionality is in a separate class because the API is experimental.

Parameters

- **cache** (*str*) – passed to QueryService, which see for explanation
- **api_key** (*str*) – API key from NCBI

Raises *EutilsError* – if cache file couldn't be created

fetch_gbseq_by_ac (*acv*)

fetch_gene_by_hgnc (*hgnc*)

fetch_nuccore_by_ac (*acv*)

fetch_snps_for_gene (*hgnc, organism='human'*)

1.4 Contributing

Development occurs in the default branch. Please work in feature branches or bookmarks from the default branch. Feature branches should be named for the utils issue they fix, as in *121-update-xml-facades*. When merging, use a commit message like “closes #121: update xml facades to new-style interface”. (“closes #n” is recognized automatically and closes the ticket upon pushing.)

The included Makefile automates many tasks. In particular, *make develop* prepares a development environment and *make test* runs unittests. (Please run tests before committing!)

Again, thanks for your contributions.

1.4.1 Development

This section is intended for developers seeking to extend the utils package. You should be familiar with the architecture, conventions, and basic functionality elsewhere in this documentation.

Get Cozy with make

The utils package includes a GNU Makefile that aids nearly all developer tasks. It subsumes much of the functionality in setup.py. While using the Makefile isn't required to develop, it is the official way to invoke tools, tests, and other development features. Type *make* for utils-specific help.

Some of the key targets are:

develop Prepare the directory for local development.

install Install utils (as with python setup.py install).

test Run the default test suite

clean, cleaner, cleanest Remove extraneous files, leaving a directory in various states of tidiness.

docs Make the sphinx docs in doc/build/html/.

upload, upload_docs Upload package to PyPI or docs to pythonhosted.org.

Installation for Development

You will need [Mercurial](#) to clone the utils repository.

```
$ hg clone ssh://hg@bitbucket.org/biocommons/utils/
$ cd utils
$ make develop
```

Submitting Patches

Yes! We'll be thrilled to have your contributions!

The preferred way to submit a patch is by forking the project on BitBucket, committing your changes there, then sending a pull request.

If you have a really worthwhile patch, we'll probably accept a diff-formatted patch, but that'll make it harder for us and impossible for you to get credit.

Developing and Contributing to utils

- Fork the project at <https://bitbucket.org/biocommons/utils/>
- Clone the project locally with:

```
$ hg clone https://bitbucket.org/<your_username>/eutils
```

- Create a virtualenv:

```
$ mkvirtualenv eutils
```

mkvirtualenv is part of the virtualenvwrapper package. Python3 users should prefer pyenv.

- Prepare your environment:

```
$ make develop
```

(The Makefile in eutils wraps functionality in setup.py, and also provides many useful utilitarian rules. Type `make` to see a list of targets.)

- Create an issue at <https://github.com/biocommons/eutils/issues/> for the feature you want to work on. This helps tracking for changelogs.
- Create a mercurial bookmark for your feature. Please name the bookmark like 141-implement-caching (where 141 is the issue number).
- Code away, then commit and push:

```
$ hg commit -m 'fixes #141: implemented caching'

$ hg push
```

- If you'd like to contribute back, create a pull request.

1.5 ChangeLog

1.5.1 0.6 Series

0.6.0 (2019-12-17)

Changes since 0.5.2 (2019-05-15).

Special Attention

Support for Python 2.7 has been dropped.

Deprecations

Bug Fixes

- Closes [#164](#): Handle cases of NCBI error response without ERROR message [[2e55029](#)] (Reece Hart)

New Features

Other Changes

- Replace deprecated time.clock function (#165) [caeff2b] (Jeff van Santen)
- Replaced process_time() with monotonic(), which measures elapsed time [d4508f8] (Reece Hart)

Internal and Developer Changes

- Add .deepsources.toml, Fix issues raised by DeepSource (#163) [db41c8b] (Mohit Solanki)
- Dropped python 2.7 support [3e53fa2] (Reece Hart)
- Closes #166: Remove six and _internal/compat.py [b5d4e84] (Reece Hart)
- Sync with current biocommons conventions: use setup.cfg, use src/, Makefile updates, drop req files [a232d9b] (Reece Hart)
- Add support for testing on Python 3.8 on travis and tox [0e6e2f0] (Reece Hart)

1.5.2 0.5 Series

0.5.0 (2018-11-19)

Changes since 0.4.1 (2018-10-10).

Special Attention

Support for Python < 3.6 will be dropped on 2019-03-31. This release will issue warnings when imported by a python version < 3.6. See <https://github.com/biocommons/org/wiki/Migrating-to-Python-3.6>

New Features

- #159: Adding publication types, mesh qualifiers and chemicals properties for medline citations [d28dbd0]. Thanks to @VincentMatthys for this contribution!

Internal and Developer Changes

- removed py 3.5 from travis config; updated badges in README [5ab476c]
- sync project files with bioutils; use travis stages for test+deploy [2a2222e]
- deprecate python < 3.6; see <https://github.com/biocommons/org/wiki/Migrating-to-Python-3.6> [57c41cc]

1.5.3 0.4 Series

Important: Clients should import only from eutils (e.g., `import eutils`). Importing from submodules (e.g., `import eutils.client`) will fail in the 0.5 release.

0.4.1 (2018-10-11)

Changes since 0.4.0 (2018-05-28).

Bug Fixes

- closes #157: edge case results in error being thrown when extracting abstract text (#158) [@pmartin23]

0.4.0 (2018-05-28)

Changes since 0.3.2 (2017-11-02).

Special Attention

- fixes #152: support NCBI API keys [1175630]

Removed time-based throttling and added support for NCBI API keys. Without an API key, NCBI returns an error if clients make more than 3 requests/second. With an API, clients may make 10 requests/second. See <https://www.ncbi.nlm.nih.gov/books/NBK25497/>. The API key may be passed as argument to Client and QueryService.

New Features

- #153: efetch for protein db

Internal and Developer Changes

- closes #132: Fix functions default argument (#156)

1.5.4 0.3 Series

0.3.2 (2017-11-02)

Changes since 0.3.1 (2017-08-22).

Internal and Developer Changes

- Fix for #150 drop gi assertion in tests [e857856]

0.3.1 (2017-08-22)

Changes since 0.3.0.post0 (2017-03-01).

Bug Fixes

- closes #149: fix bug in which multi-line abstracts were truncated [df491f8]

0.3.0 (2017-02-28)

Changes since 0.2.4 (2017-02-21).

Special Attention

- closes [#147](#): Do not use cache by default (use `cache=True` for current behavior) [[abfa129](#)]

Internal and Developer Changes

- closes [#140](#): migrate tests to vcrpy (and delete test data) [[7862548](#)]

1.5.5 0.2 Series

0.2.4 (2017-02-21)

Changes since 0.2.3 (2017-02-03).

Special Attention

Deprecations

Bug Fixes

New Features

Other Changes

Internal and Developer Changes

- Implemented wheel support

0.2.3 (2017-02-03)

Changes since 0.2.2 (2016-12-10).

New Features

- Merge pull request [#143](#) from diekhans/betterResponseParseErrors
- Merge pull request [#144](#) from diekhans/nucestSupport [[f26636c](#)]

Other Changes

- include error message from XML parser when can't parse NCBI response [[47d0673](#)]

Internal and Developer Changes

- use exception syntax compatible with py 3 [858cbdc]

0.2.2 (2016-12-10)

Changes since 0.2.1 (2016-10-19).

Internal and Developer Changes

- closes #141: updates issue links from bitbucket to github [c41c501]

0.2.1 (2016-10-19)

Changes since 0.2.0 (2016-09-15).

Bug Fixes

- Merge pull request #139 from timothyjlaurent/138-catch-malformed-xml-on-error

0.2.0 (2016-09-15)

Changes since 0.1.1 (2016-07-22).

New Features

- fixes #8: use https for eutilities (required by 2016-09-30) [0ce382f]
- Update ExchangeSet handling for switch to https [399b330]
- basic support for pubmedcentral [18b3a63]

1.6 License

eutils is released under the [Apache License 2.0](#), the text of which appears below:

```

        Apache License
        Version 2.0, January 2004
        http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by

```

(continues on next page)

(continued from previous page)

the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of,

(continues on next page)

(continued from previous page)

publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
 - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use,

(continues on next page)

(continued from previous page)

reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate	
---	--

(continues on next page)

(continued from previous page)

```
comment syntax for the file format. We also recommend that a
file or class name and description of purpose be included on the
same "printed page" as the copyright notice for easier
identification within third-party archives.
```

```
Copyright [yyyy] [name of copyright owner]
```

```
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```


CHAPTER 2

Indices and tables

- `genindex`
- `modindex`

e

`eutils.sketchy.clientx`, [7](#)

C

`Client` (class in *eutils*), 4
`ClientX` (class in *eutils.sketchy.clientx*), 7

D

databases (*eutils.Client* attribute), 4

E

`efetch()` (*eutils.Client* method), 4
`efetch()` (*eutils.QueryService* method), 5
`einfo()` (*eutils.Client* method), 4
`einfo()` (*eutils.QueryService* method), 5
`elink()` (*eutils.QueryService* method), 6
`esearch()` (*eutils.Client* method), 4
`esearch()` (*eutils.QueryService* method), 6
`esummary()` (*eutils.QueryService* method), 6
`eutils.sketchy.clientx` (module), 7
`EutilsError` (class in *eutils*), 7
`EutilsNCBIError` (class in *eutils*), 7
`EutilsNotFoundError` (class in *eutils*), 7
`EutilsRequestError` (class in *eutils*), 7

F

`fetch_gbseq_by_ac()` (*eutils.sketchy.clientx.ClientX* method), 7
`fetch_gene_by_hgnc()` (*eutils.sketchy.clientx.ClientX* method), 7
`fetch_nuccore_by_ac()` (*eutils.sketchy.clientx.ClientX* method), 7
`fetch_snps_for_gene()` (*eutils.sketchy.clientx.ClientX* method), 7

Q

`QueryService` (class in *eutils*), 4