Elasticsearch Documentation

Release 0.4.5

Honza Král

March 08, 2014

Official low-level client for Elasticsearch. It's goal is to provide common ground for all Elasticsearch-related code in Python; because of this it tries to be opinion-free and very extendable.

CHAPTER 1

Example Usage

```
from elasticsearch import Elasticsearch
es = Elasticsearch()

doc = {
    'author': 'kimchy',
    'text': 'Elasticsearch: cool. bonsai cool.',
    'timestamp': datetime(2010, 10, 10, 10, 10, 10)
}
res = es.index(index="test-index", doc_type='tweet', id=1, body=doc)
print(res['ok'])

res = es.get(index="test-index", doc_type='tweet', id=1)
print(res['_source'])

es.indices.refresh(index="test-index")

res = es.search(index="test-index", body={"query": {"match_all": {}}})
print("Got %d Hits:" % res['hits']['total'])
for hit in res['hits']['hits']:
    print("%(timestamp)s %(author)s: %(text)s" % hit["_source"])
```

CHAPTER 2

Features

This client was designed as very thin wrapper around Elasticseach's REST API to allow for maximum flexibility. This means that there are no opinions in this client; it also means that some of the APIs are a little cumbersome to use from Python. We have created some *Helpers* to help with this issue.

2.1 Persistent Connections

`elasticsearch-py` uses persistent connections inside of individual connection pools (one per each configured or sniffed node). Out of the box you can choose to use `http`, `thrift` or an experimental `memcached` protocol to communicate with the elasticsearch nodes. See *Transport classes* for more information.

The transport layer will create an instance of the selected connection class per node and keep track of the health of individual nodes - if a node becomes unresponsive (throwing exceptions while connecting to it) it's put on a timeout by the `ConnectionPool` class and only returned to the circulation after the timeout is over (or when no live nodes are left). By default node are randomized before passed into the pool and round-robin strategy is used for load balancing.

You can customize this behavior by passing parameters to the *Connection Layer API* (all keyword arguments to the `Elasticsearch` class will be passed through). If what you want to accomplish is not supported you should be able to create a subclass of the relevant component and pass it in as a parameter to be used instead of the default implementation.

2.2 Sniffing

The client can be configured to inspect the cluster state to get a list of nodes upon startup, periodically and/or on failure. See `Transport` parameters for details.

Some example configurations:

```
from elasticsearch import Elasticsearch

# by default we don't sniff, ever
es = Elasticsearch()

# you can specify to sniff on startup to inspect the cluster and load
# balance across all nodes
es = Elasticsearch(["seed1", "seed2"], sniff_on_start=True)

# you can also sniff periodically and/or after failure:
es = Elasticsearch(["seed1", "seed2"], sniff_on_start=True, sniff_on_connection_fail=True, sniffer_t
```

2.3 Logging

`elasticsearch-py` uses the standard logging library from python to define two loggers: `elasticsearch` and `elasticsearch.trace`. `elasticsearch` is used by the client to log standard activity, depending on the log level. `elasticsearch.trace` can be used to log requests to the server in the form of `curl` commands using pretty-printed json that can then be executed from command line. The tace logger doesn't inherit from the base one - it needs to be activated separately.

CHAPTER 3

Contents

3.1 API Documentation

Note: All the API calls map the raw REST api as closely as possible, including the distinction between required and optional arguments to the calls. This means that the code makes distinction between positional and keyword arguments; we, however, recommend that people use keyword arguments for all calls for consistency and safety.

Note: for compatibility with the Python ecosystem we use `from_` instead of `from` and `doc_type` instead of `type` as parameter names.

3.1.1 Elasticsearch

class `elasticsearch.`**`Elasticsearch`**(*hosts=None*, *transport_class=<class 'elasticsearch.transport.Transport'>*, ***kwargs*)

Elasticsearch low-level client. Provides a straightforward mapping from Python to ES REST endpoints.

The instance has attributes *indices* and *cluster* that provide access to `IndicesClient` and `ClusterClient` instances respectively.

Parameters

- **hosts** – list of nodes we should connect to. Node should be a dictionary ({"host": "localhost", "port": 9200}), the entire dictionary will be passed to the `Connection` class as kwargs, or a string in the format ot `host[:port]` which will be translated to a dictionary automatically. If no value is given the `Connection` class defaults will be used.
- **transport_class** – `Transport` subclass to use.
- **kwargs** – any additional arguments will be passed on to the `Transport` class and, subsequently, to the `Connection` instances.

`bulk`(**args*, ***kwargs*)

Perform many index/delete operations in a single API call. http://elasticsearch.org/guide/reference/api/bulk/

See the `bulk()` helper function for a more friendly API.

Parameters

- **body** – The operation definition and data (action-data pairs), as either a newline separated string, or a sequence of dicts to serialize (one per row).
- **index** – Default index for items which don't provide one

- **doc_type** – Default document type for items which don't provide one
- **consistency** – Explicit write consistency setting for the operation
- **refresh** – Refresh the index after performing the operation
- **replication** – Explicitly set the replication type (default: sync)
- **timeout** – Explicit operation timeout

clear_scroll(**args*, ***kwargs*)

Clear the scroll request created by specifying the scroll parameter to search. http://www.elasticsearch.org/guide/reference/api/search/scroll/

Parameters **scroll_id** – The scroll ID or a list of scroll IDs

count(**args*, ***kwargs*)

Execute a query and get the number of matches for that query. http://elasticsearch.org/guide/reference/api/count/

Parameters

- **index** – A comma-separated list of indices to restrict the results
- **doc_type** – A comma-separated list of types to restrict the results
- **body** – A query to restrict the results (optional)
- **ignore_indices** – When performed on multiple indices, allows to ignore *missing* ones (default: none)
- **min_score** – Include only documents with a specific *_score* value in the result
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **q** – Query in the Lucene query string syntax
- **routing** – Specific routing value
- **source** – The URL-encoded query definition (instead of using the request body)

create(**args*, ***kwargs*)

Adds a typed JSON document in a specific index, making it searchable. Behind the scenes this method calls index(..., op_type='create') http://elasticsearch.org/guide/reference/api/index_/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document
- **id** – Document ID
- **body** – The document
- **consistency** – Explicit write consistency setting for the operation
- **id** – Specific document ID (when the POST method is used)
- **parent** – ID of the parent document
- **percolate** – Percolator queries to execute while indexing the document
- **refresh** – Refresh the index after performing the operation
- **replication** – Specific replication type (default: sync)
- **routing** – Specific routing value

- **timeout** – Explicit operation timeout
- **timestamp** – Explicit timestamp for the document
- **ttl** – Expiration time for the document
- **version** – Explicit version number for concurrency control
- **version_type** – Specific version type

delete(**args*, ***kwargs*)
Delete a typed JSON document from a specific index based on its id. http://elasticsearch.org/guide/reference/api/delete/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document
- **id** – The document ID
- **consistency** – Specific write consistency setting for the operation
- **parent** – ID of parent document
- **refresh** – Refresh the index after performing the operation
- **replication** – Specific replication type (default: sync)
- **routing** – Specific routing value
- **timeout** – Explicit operation timeout
- **version** – Explicit version number for concurrency control
- **version_type** – Specific version type

delete_by_query(**args*, ***kwargs*)
Delete documents from one or more indices and one or more types based on a query. http://www.elasticsearch.org/guide/reference/api/delete-by-query/

Parameters

- **index** – A comma-separated list of indices to restrict the operation
- **doc_type** – A comma-separated list of types to restrict the operation
- **body** – A query to restrict the operation
- **consistency** – Specific write consistency setting for the operation
- **ignore_indices** – When performed on multiple indices, allows to ignore *missing* ones (default: none)
- **replication** – Specific replication type (default: sync)
- **routing** – Specific routing value
- **source** – The URL-encoded query definition (instead of using the request body)
- **q** – Query in the Lucene query string syntax
- **timeout** – Explicit operation timeout

exists(**args*, ***kwargs*)
Returns a boolean indicating whether or not given document exists in Elasticsearch. http://elasticsearch.org/guide/reference/api/get/

Parameters

- **index** – The name of the index
- **id** – The document ID
- **doc_type** – The type of the document (uses *_all* by default to fetch the first document matching the ID across all types)
- **parent** – The ID of the parent document
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **realtime** – Specify whether to perform the operation in realtime or search mode
- **refresh** – Refresh the shard containing the document before performing the operation
- **routing** – Specific routing value

explain(**args*, ***kwargs*)

The explain api computes a score explanation for a query and a specific document. This can give useful feedback whether a document matches or didn't match a specific query. http://elasticsearch.org/guide/reference/api/explain/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document
- **id** – The document ID
- **body** – The query definition using the Query DSL
- **_source** – True or false to return the _source field or not, or a list of fields to return
- **_source_exclude** – A list of fields to exclude from the returned _source field
- **_source_include** – A list of fields to extract and return from the _source field
- **analyze_wildcard** – Specify whether wildcards and prefix queries in the query string query should be analyzed (default: false)
- **analyzer** – The analyzer for the query string query
- **default_operator** – The default operator for query string query (AND or OR), (default: OR)
- **df** – The default field for query string query (default: _all)
- **fields** – A comma-separated list of fields to return in the response
- **lenient** – Specify whether format-based query failures (such as providing text to a numeric field) should be ignored
- **lowercase_expanded_terms** – Specify whether query terms should be lowercased
- **parent** – The ID of the parent document
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **q** – Query in the Lucene query string syntax
- **routing** – Specific routing value
- **source** – The URL-encoded query definition (instead of using the request body)

get(**args*, ***kwargs*)
Get a typed JSON document from the index based on its id. http://elasticsearch.org/guide/reference/api/get/

Parameters

- **index** – The name of the index
- **id** – The document ID
- **doc_type** – The type of the document (uses *_all* by default to fetch the first document matching the ID across all types)
- **_source** – True or false to return the _source field or not, or a list of fields to return
- **_source_exclude** – A list of fields to exclude from the returned _source field
- **_source_include** – A list of fields to extract and return from the _source field
- **fields** – A comma-separated list of fields to return in the response
- **parent** – The ID of the parent document
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **realtime** – Specify whether to perform the operation in realtime or search mode
- **refresh** – Refresh the shard containing the document before performing the operation
- **routing** – Specific routing value

get_source(**args*, ***kwargs*)
Get the source of a document by it's index, type and id. http://elasticsearch.org/guide/reference/api/get/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document (uses *_all* by default to fetch the first document matching the ID across all types)
- **id** – The document ID
- **exclude** – A list of fields to exclude from the returned _source field
- **include** – A list of fields to extract and return from the _source field
- **parent** – The ID of the parent document
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **realtime** – Specify whether to perform the operation in realtime or search mode
- **refresh** – Refresh the shard containing the document before performing the operation
- **routing** – Specific routing value

index(**args*, ***kwargs*)
Adds or updates a typed JSON document in a specific index, making it searchable. http://elasticsearch.org/guide/reference/api/index_/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document

- **body** – The document
- **id** – Document ID
- **consistency** – Explicit write consistency setting for the operation
- **op_type** – Explicit operation type (default: index)
- **parent** – ID of the parent document
- **percolate** – Percolator queries to execute while indexing the document
- **refresh** – Refresh the index after performing the operation
- **replication** – Specific replication type (default: sync)
- **routing** – Specific routing value
- **timeout** – Explicit operation timeout
- **timestamp** – Explicit timestamp for the document
- **ttl** – Expiration time for the document
- **version** – Explicit version number for concurrency control
- **version_type** – Specific version type

info(**args*, ***kwargs*)
Get the basic info from the current cluster.

mget(**args*, ***kwargs*)
Get multiple documents based on an index, type (optional) and ids. http://elasticsearch.org/guide/reference/api/multi-get/

Parameters

- **body** – Document identifiers; can be either *docs* (containing full document information) or *ids* (when index and type is provided in the URL.
- **index** – The name of the index
- **doc_type** – The type of the document
- **_source** – True or false to return the _source field or not, or a list of fields to return
- **_source_exclude** – A list of fields to exclude from the returned _source field
- **_source_include** – A list of fields to extract and return from the _source field
- **fields** – A comma-separated list of fields to return in the response
- **parent** – The ID of the parent document
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **realtime** – Specify whether to perform the operation in realtime or search mode
- **refresh** – Refresh the shard containing the document before performing the operation
- **routing** – Specific routing value

mlt(**args*, ***kwargs*)
Get documents that are "like" a specified document. http://elasticsearch.org/guide/reference/api/more-like-this/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document (use *_all* to fetch the first document matching the ID across all types)
- **id** – The document ID
- **body** – A specific search request definition
- **boost_terms** – The boost factor
- **max_doc_freq** – The word occurrence frequency as count: words with higher occurrence in the corpus will be ignored
- **max_query_terms** – The maximum query terms to be included in the generated query
- **max_word_len** – The minimum length of the word: longer words will be ignored
- **min_doc_freq** – The word occurrence frequency as count: words with lower occurrence in the corpus will be ignored
- **min_term_freq** – The term frequency as percent: terms with lower occurence in the source document will be ignored
- **min_word_len** – The minimum length of the word: shorter words will be ignored
- **mlt_fields** – Specific fields to perform the query against
- **percent_terms_to_match** – How many terms have to match in order to consider the document a match (default: 0.3)
- **routing** – Specific routing value
- **search_from** – The offset from which to return results
- **search_indices** – A comma-separated list of indices to perform the query against (default: the index containing the document)
- **search_query_hint** – The search query hint
- **search_scroll** – A scroll search request definition
- **search_size** – The number of documents to return (default: 10)
- **search_source** – A specific search request definition (instead of using the request body)
- **search_type** – Specific search type (eg. *dfs_then_fetch*, *count*, etc)
- **search_types** – A comma-separated list of types to perform the query against (default: the same type as the document)
- **stop_words** – A list of stop words to be ignored

msearch(**args*, ***kwargs*)

Execute several search requests within the same API. http://www.elasticsearch.org/guide/reference/api/multi-search/

Parameters

- **body** – The request definitions (metadata-search request definition pairs), as either a newline separated string, or a sequence of dicts to serialize (one per row).
- **index** – A comma-separated list of index names to use as default
- **doc_type** – A comma-separated list of document types to use as default
- **search_type** – Search operation type

percolate(**args*, ***kwargs*)
Send a percolate request which include a doc, and get back the queries that match on that doc out of the set of registered queries. http://elasticsearch.org/guide/reference/api/percolate/

Parameters

- **index** – The name of the index with a registered percolator query
- **doc_type** – The document type
- **body** – The document (*doc*) to percolate against registered queries; optionally also a *query* to limit the percolation to specific registered queries
- **prefer_local** – With *true*, specify that a local shard should be used if available, with *false*, use a random shard (default: true)

ping(**args*, ***kwargs*)
Returns True if the cluster is up, False otherwise.

scroll(**args*, ***kwargs*)
Scroll a search request created by specifying the scroll parameter. http://www.elasticsearch.org/guide/reference/api/search/scroll/

Parameters

- **scroll_id** – The scroll ID
- **scroll** – Specify how long a consistent view of the index should be maintained for scrolled search

search(**args*, ***kwargs*)
Execute a search query and get back search hits that match the query. http://www.elasticsearch.org/guide/reference/api/search/

Parameters

- **index** – A comma-separated list of index names to search; use *_all* or empty string to perform the operation on all indices
- **doc_type** – A comma-separated list of document types to search; leave empty to perform the operation on all types
- **body** – The search definition using the Query DSL
- **_source** – True or false to return the _source field or not, or a list of fields to return
- **_source_exclude** – A list of fields to exclude from the returned _source field
- **_source_include** – A list of fields to extract and return from the _source field
- **analyze_wildcard** – Specify whether wildcard and prefix queries should be analyzed (default: false)
- **analyzer** – The analyzer to use for the query string
- **default_operator** – The default operator for query string query (AND or OR) (default: OR)
- **df** – The field to use as default where no field prefix is given in the query string
- **explain** – Specify whether to return detailed information about score computation as part of a hit
- **fields** – A comma-separated list of fields to return as part of a hit

- **ignore_indices** – When performed on multiple indices, allows to ignore *missing* ones (default: none)
- **indices_boost** – Comma-separated list of index boosts
- **lenient** – Specify whether format-based query failures (such as providing text to a numeric field) should be ignored
- **lowercase_expanded_terms** – Specify whether query terms should be lowercased
- **from** – Starting offset (default: 0)
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **q** – Query in the Lucene query string syntax
- **routing** – A comma-separated list of specific routing values
- **scroll** – Specify how long a consistent view of the index should be maintained for scrolled search
- **search_type** – Search operation type
- **size** – Number of hits to return (default: 10)
- **sort** – A comma-separated list of <field>:<direction> pairs
- **source** – The URL-encoded request definition using the Query DSL (instead of using request body)
- **stats** – Specific 'tag' of the request for logging and statistical purposes
- **suggest_field** – Specify which field to use for suggestions
- **suggest_mode** – Specify suggest mode (default: missing)
- **suggest_size** – How many suggestions to return in response
- **suggest_text** – The source text for which the suggestions should be returned
- **timeout** – Explicit operation timeout
- **version** – Specify whether to return document version as part of a hit

suggest(**args*, ***kwargs*)

The suggest feature suggests similar looking terms based on a provided text by using a suggester. http://elasticsearch.org/guide/reference/api/search/suggest/

Parameters

- **index** – A comma-separated list of index names to restrict the operation; use *_all* or empty string to perform the operation on all indices
- **body** – The request definition
- **ignore_indices** – When performed on multiple indices, allows to ignore *missing* ones (default: none)
- **preference** – Specify the node or shard the operation should be performed on (default: random)
- **routing** – Specific routing value
- **source** – The URL-encoded request definition (instead of using request body)

update(**args*, ***kwargs*)
Update a document based on a script or partial data provided. http://elasticsearch.org/guide/reference/api/update/

Parameters

- **index** – The name of the index
- **doc_type** – The type of the document
- **id** – Document ID
- **body** – The request definition using either *script* or partial *doc*
- **consistency** – Explicit write consistency setting for the operation
- **fields** – A comma-separated list of fields to return in the response
- **lang** – The script language (default: mvel)
- **parent** – ID of the parent document
- **percolate** – Perform percolation during the operation; use specific registered query name, attribute, or wildcard
- **refresh** – Refresh the index after performing the operation
- **replication** – Specific replication type (default: sync)
- **retry_on_conflict** – Specify how many times should the operation be retried when a conflict occurs (default: 0)
- **routing** – Specific routing value
- **script** – The URL-encoded script definition (instead of using request body)
- **timeout** – Explicit operation timeout
- **timestamp** – Explicit timestamp for the document
- **ttl** – Expiration time for the document
- **version** – Explicit version number for concurrency control
- **version_type** – Explicit version number for concurrency control

3.1.2 Indices

class elasticsearch.client.**IndicesClient**(*client*)

analyze(**args*, ***kwargs*)
Perform the analysis process on a text and return the tokens breakdown of the text. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-analyze.html

Parameters

- **index** – The name of the index to scope the operation
- **body** – The text on which the analysis should be performed
- **analyzer** – The name of the analyzer to use
- **field** – Use the analyzer configured for this field (instead of passing the analyzer name)
- **filters** – A comma-separated list of filters to use for the analysis

- **format** – Format of the output, default u'detailed'
- **index** – The name of the index to scope the operation
- **prefer_local** – With *true*, specify that a local shard should be used if available, with *false*, use a random shard (default: true)
- **text** – The text on which the analysis should be performed (when request body is not used)
- **tokenizer** – The name of the tokenizer to use for the analysis

clear_cache(**args*, ***kwargs*)

Clear either all caches or specific cached associated with one ore more indices. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-clearcache.html

Parameters

- **index** – A comma-separated list of index name to limit the operation
- **field_data** – Clear field data
- **fielddata** – Clear field data
- **fields** – A comma-separated list of fields to clear when using the *field_data* parameter (default: all)
- **filter** – Clear filter caches
- **filter_cache** – Clear filter caches
- **filter_keys** – A comma-separated list of keys to clear when using the *filter_cache* parameter (default: all)
- **id** – Clear ID caches for parent/child
- **id_cache** – Clear ID caches for parent/child
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones (default: none)

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)
- **index** – A comma-separated list of index name to limit the operation
- **recycler** – Clear the recycler cache

close(**args*, ***kwargs*)

Close an index to remove it's overhead from the cluster. Closed index is blocked for read/write operations. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-open-close.html

Parameters

- **index** – The name of the index
- **master_timeout** – Specify timeout for connection to master

- **timeout** – Explicit operation timeout

create(**args*, ***kwargs*)
Create an index in Elasticsearch. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-create-index.html

Parameters

- **index** – The name of the index
- **body** – The configuration for the index (*settings* and *mappings*)
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

delete(**args*, ***kwargs*)
Delete an index in Elasticsearch http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-delete-index.html

Parameters

- **index** – A comma-separated list of indices to delete; use *_all* or empty string to delete all indices
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

delete_alias(**args*, ***kwargs*)
Delete specific alias. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html

Parameters

- **index** – The name of the index with an alias
- **name** – The name of the alias to be deleted
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit timestamp for the document

delete_mapping(**args*, ***kwargs*)
Delete a mapping (type) along with its data. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-delete-mapping.html

Parameters

- **index** – A comma-separated list of index names; use *_all* for all indices
- **doc_type** – The name of the document type to delete
- **master_timeout** – Specify timeout for connection to master

delete_template(**args*, ***kwargs*)
Delete an index template by its name. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-templates.html

Parameters

- **name** – The name of the template
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

delete_warmer(**args*, ***kwargs*)
Delete an index warmer. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-warmers.html

Parameters

- **index** – A comma-separated list of index names to register warmer for; use *_all* or empty string to perform the operation on all indices
- **doc_type** – A comma-separated list of document types to register warmer for; use *_all* or empty string to perform the operation on all types
- **name** – The name of the warmer (supports wildcards); leave empty to delete all warmers
- **master_timeout** – Specify timeout for connection to master

exists(**args*, ***kwargs*)
Return a boolean indicating whether given index exists. http://www.elasticsearch.org/guide/en/elasticsearch/reference/curren indices-exists.html

Parameters **index** – A list of indices to check

exists_alias(**args*, ***kwargs*)
Return a boolean indicating whether given alias exists. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current aliases.html

Parameters

- **name** – A comma-separated list of alias names to return
- **index** – A comma-separated list of index names to filter aliases
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones (default: none)

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

exists_type(**args*, ***kwargs*)
Check if a type/types exists in an index/indices. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indice types-exists.html

Parameters

- **index** – A comma-separated list of index names; use *_all* to check the types across all indices
- **doc_type** – A comma-separated list of document types to check
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

arg ignore_indices When performed on multiple indices, allows to

ignore *missing* ones (default: none)

arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

flush(**args*, ***kwargs*)

Explicitly flush one or more indices. http://http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-flush.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string for all indices
- **force** – Whether a flush should be forced even if it is not necessarily needed ie. if no changes will be committed to the index.
- **full** – If set to true a new index writer is created and settings that have been changed related to the index writer will be refreshed.
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones (default: none)

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

- **refresh** – Refresh the index after performing the operation

get_alias(**args*, ***kwargs*)

Retrieve a specified alias. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html

Parameters

- **name** – A comma-separated list of alias names to return
- **index** – A comma-separated list of index names to filter aliases
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

arg ignore_indices When performed on multiple indices, allows to

ignore *missing* ones, default u'none'

arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

get_aliases(**args*, ***kwargs*)
Retrieve specified aliases http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html

Parameters

- **index** – A comma-separated list of index names to filter aliases
- **timeout** – Explicit operation timeout

get_field_mapping(**args*, ***kwargs*)
Retrieve mapping definition of a specific field. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-get-field-mapping.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string for all indices
- **doc_type** – A comma-separated list of document types
- **field** – A comma-separated list of fields to retrieve the mapping for
- **include_defaults** – A boolean indicating whether to return default values

get_mapping(**args*, ***kwargs*)
Retrieve mapping definition of index or index/type. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-get-mapping.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string for all indices
- **doc_type** – A comma-separated list of document types

get_settings(**args*, ***kwargs*)
Retrieve settings for one or more (or all) indices. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-get-settings.html

Parameters **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices

get_template(**args*, ***kwargs*)
Retrieve an index template by its name. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-templates.html

Parameters **name** – The name of the template

get_warmer(**args*, ***kwargs*)
Retreieve an index warmer. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-warmers.html

Parameters

- **index** – A comma-separated list of index names to restrict the operation; use *_all* to perform the operation on all indices
- **doc_type** – A comma-separated list of document types to restrict the operation; leave empty to perform the operation on all types
- **name** – The name of the warmer (supports wildcards); leave empty to get all warmers

open(**args*, ***kwargs*)

Open a closed index to make it available for search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/in open-close.html

Parameters

- **index** – The name of the index
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

optimize(**args*, ***kwargs*)

Explicitly optimize one or more indices through an API. http://www.elasticsearch.org/guide/en/elasticsearch/reference/curre optimize.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **flush** – Specify whether the index should be flushed after performing the operation (default: true)
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones, default u'none'

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)
- **max_num_segments** – The number of segments the index should be merged into (default: dynamic)
- **only_expunge_deletes** – Specify whether the operation should only expunge deleted documents
- **operation_threading** – TODO: ?
- **refresh** – Specify whether the index should be refreshed after performing the operation (default: true)
- **wait_for_merge** – Specify whether the request should block until the merge process is finished (default: true)

put_alias(**args*, ***kwargs*)
Create an alias for a specific index/indices. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html

Parameters

- **index** – The name of the index with an alias
- **name** – The name of the alias to be created or updated
- **body** – The settings for the alias, such as *routing* or *filter*
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit timestamp for the document

put_mapping(**args*, ***kwargs*)
Register specific mapping definition for a specific type. http://www.elasticsearch.org/guide/en/elasticsearch/reference/curren put-mapping.html

Parameters

- **index** – A comma-separated list of index names; use *_all* to perform the operation on all indices
- **doc_type** – The name of the document type
- **body** – The mapping definition
- **ignore_conflicts** – Specify whether to ignore conflicts while updating the mapping (default: false)
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

put_settings(**args*, ***kwargs*)
Change specific index level settings in real time. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indic update-settings.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **master_timeout** – Specify timeout for connection to master
- **body** – The index settings to be updated

put_template(**args*, ***kwargs*)
Create an index template that will automatically be applied to new indices created. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-templates.html

Parameters

- **name** – The name of the template
- **body** – The template definition
- **order** – The order for this template when merging multiple matching ones (higher numbers are merged later, overriding the lower numbers)
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Explicit operation timeout

put_warmer(**args*, ***kwargs*)
Create an index warmer to run registered search requests to warm up the index before it is available for search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-warmers.html

Parameters

- **index** – A comma-separated list of index names to register the warmer for; use *_all* or empty string to perform the operation on all indices
- **name** – The name of the warmer
- **doc_type** – A comma-separated list of document types to register the warmer for; leave empty to perform the operation on all types
- **body** – The search request definition for the warmer (query, filters, facets, sorting, etc)
- **master_timeout** – Specify timeout for connection to master

refresh(**args*, ***kwargs*)
Explicitly refresh one or more index, making all operations performed since the last refresh available for search. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-refresh.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones, default u'none'

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

segments(**args*, ***kwargs*)
Provide low level segments information that a Lucene index (shard level) is built with. http://elasticsearch.org/guide/reference/api/admin-indices-segments/

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

arg ignore_indices When performed on multiple indices, allows to

ignore *missing* ones, default u'none'

arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

- **operation_threading** – TODO: ?

snapshot_index(**args*, ***kwargs*)

Explicitly perform a snapshot through the gateway of one or more indices (backup them). http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-gateway-snapshot.html

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string for all indices
- **allow_no_indices** –

Whether to ignore if a wildcard indices

expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

arg ignore_indices When performed on multiple indices, allows to

ignore *missing* ones (default: none)

arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

stats(**args*, ***kwargs*)

Retrieve statistics on different operations happening on an index. http://elasticsearch.org/guide/reference/api/admin-indices-stats/

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **metric_family** – Limit the information returned to a specific metric
- **all** – Return all available information
- **clear** – Reset the default level of detail
- **completion** – Return information about completion suggester stats
- **completion_fields** – A comma-separated list of fields for *completion* metric (supports wildcards)
- **docs** – Return information about indexed and deleted documents
- **fielddata** – Return information about field data
- **fielddata_fields** – A comma-separated list of fields for *fielddata* metric (supports wildcards)
- **fields** – A comma-separated list of fields for *fielddata* and *completion* metric (supports wildcards)

- **filter_cache** – Return information about filter cache
- **flush** – Return information about flush operations
- **get** – Return information about get operations
- **groups** – A comma-separated list of search groups for *search* statistics
- **id_cache** – Return information about ID cache
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones (default: none)

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)
- **indexing** – Return information about indexing operations
- **merge** – Return information about merge operations
- **refresh** – Return information about refresh operations
- **search** – Return information about search operations; use the *groups* parameter to include information for specific search groups
- **store** – Return information about the size of the index
- **warmer** – Return information about warmers

status(**args*, ***kwargs*)

Get a comprehensive status information of one or more indices. http://elasticsearch.org/guide/reference/api/admin-indices-_/

Parameters

- **index** – A comma-separated list of index names; use *_all* or empty string to perform the operation on all indices
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones, default u'none'

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)

- **operation_threading** – TODO: ?
- **recovery** – Return information about shard recovery
- **snapshot** – TODO: ?

update_aliases(**args*, ***kwargs*)
Update specified aliases. http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-aliases.html

Parameters

- **body** – The definition of *actions* to perform
- **master_timeout** – Specify timeout for connection to master
- **timeout** – Request timeout

validate_query(**args*, ***kwargs*)
Validate a potentially expensive query without executing it. http://www.elasticsearch.org/guide/reference/api/validate/

Parameters

- **index** – A comma-separated list of index names to restrict the operation; use *_all* or empty string to perform the operation on all indices
- **doc_type** – A comma-separated list of document types to restrict the operation; leave empty to perform the operation on all types
- **body** – The query definition
- **explain** – Return detailed information about the error
- **allow_no_indices** –

  Whether to ignore if a wildcard indices

  expression resolves into no concrete indices. (This includes *_all* string or when no indices have been specified)

  arg expand_wildcards Whether to expand wildcard expression to concrete indices that are open, closed or both.

  arg ignore_indices When performed on multiple indices, allows to

  ignore *missing* ones (default: none)

  arg ignore_unavailable Whether specified concrete indices should be ignored when unavailable (missing or closed)
- **operation_threading** – TODO: ?
- **q** – Query in the Lucene query string syntax
- **source** – The URL-encoded query definition (instead of using the request body)

3.1.3 Cluster

class elasticsearch.client.**ClusterClient**(*client*)

get_settings(**args*, ***kwargs*)
Get cluster settings. http://elasticsearch.org/guide/reference/api/admin-cluster-update-settings/

health(**args*, ***kwargs*)
Get a very simple status on the health of the cluster. http://elasticsearch.org/guide/reference/api/admin-cluster-health/

Parameters

- **index** – Limit the information returned to a specific index
- **level** – Specify the level of detail for returned information, default u'cluster'
- **local** – Return local information, do not retrieve the state from master node (default: false)
- **master_timeout** – Explicit operation timeout for connection to master node
- **timeout** – Explicit operation timeout
- **wait_for_active_shards** – Wait until the specified number of shards is active
- **wait_for_nodes** – Wait until the specified number of nodes is available
- **wait_for_relocating_shards** – Wait until the specified number of relocating shards is finished
- **wait_for_status** – Wait until cluster is in a specific state, default None

node_info(**args*, ***kwargs*)
Retrieve one or more (or all) of the cluster nodes' information. http://elasticsearch.org/guide/reference/api/admin-cluster-nodes-info/

Parameters

- **node_id** – A comma-separated list of node IDs or names to limit the returned information; use *_local* to return information from the node you're connecting to, leave empty to get information from all nodes
- **all** – Return all available information
- **clear** – Reset the default settings
- **http** – Return information about HTTP
- **jvm** – Return information about the JVM
- **network** – Return information about network
- **os** – Return information about the operating system
- **plugin** – Return information about plugins
- **process** – Return information about the Elasticsearch process
- **settings** – Return information about node settings
- **thread_pool** – Return information about the thread pool
- **timeout** – Explicit operation timeout
- **transport** – Return information about transport

node_shutdown(**args*, ***kwargs*)
Shutdown one or more (or all) nodes in the cluster. http://elasticsearch.org/guide/reference/api/admin-cluster-nodes-shutdown/

Parameters

- **node_id** – A comma-separated list of node IDs or names to perform the operation on; use *_local* to perform the operation on the node you're connected to, leave empty to perform the operation on all nodes
- **delay** – Set the delay for the operation (default: 1s)
- **exit** – Exit the JVM as well (default: true)

node_stats(**args*, ***kwargs*)

Retrieve one or more (or all) of the cluster nodes statistics. http://elasticsearch.org/guide/reference/api/admin-cluster-nodes-stats/

Parameters

- **node_id** – A comma-separated list of node IDs or names to limit the returned information; use *_local* to return information from the node you're connecting to, leave empty to get information from all nodes
- **metric** – Limit the information returned for *indices* family to a specific metric
- **fields** – A comma-separated list of fields to return detailed information for, when returning the *indices* metric family (supports wildcards)
- **all** – Return all available information
- **clear** – Reset the default level of detail
- **fields** – A comma-separated list of fields for *fielddata* metric (supports wildcards)
- **fs** – Return information about the filesystem
- **http** – Return information about HTTP
- **indices** – Return information about indices
- **jvm** – Return information about the JVM
- **network** – Return information about network
- **os** – Return information about the operating system
- **process** – Return information about the Elasticsearch process
- **thread_pool** – Return information about the thread pool
- **transport** – Return information about transport

put_settings(**args*, ***kwargs*)

Update cluster wide specific settings. http://elasticsearch.org/guide/reference/api/admin-cluster-update-settings/

Parameters **body** – The settings to be updated. Can be either *transient* or *persistent* (survives cluster restart).

reroute(**args*, ***kwargs*)

Explicitly execute a cluster reroute allocation command including specific commands. http://elasticsearch.org/guide/reference/api/admin-cluster-reroute/

Parameters

- **body** – The definition of *commands* to perform (*move*, *cancel*, *allocate*)
- **dry_run** – Simulate the operation only and return the resulting state
- **filter_metadata** – Don't return cluster state metadata (default: false)

state(**args*, ***kwargs*)
Get a comprehensive state information of the whole cluster. http://elasticsearch.org/guide/reference/api/admin-cluster-state/

Parameters

- **filter_blocks** – Do not return information about blocks
- **filter_index_templates** – Do not return information about index templates
- **filter_indices** – Limit returned metadata information to specific indices
- **filter_metadata** – Do not return information about indices metadata
- **filter_nodes** – Do not return information about nodes
- **filter_routing_table** – Do not return information about shard allocation (*routing_table* and *routing_nodes*)
- **local** – Return local information, do not retrieve the state from master node (default: false)
- **master_timeout** – Specify timeout for connection to master

3.2 Connection Layer API

All of the classes reponsible for handling the connection to the Elasticsearch cluster. The default subclasses used can be overriden by passing parameters to the `Elasticsearch` class. All of the arguments to the client will be passed on to `Transport`, `ConnectionPool` and `Connection`.

For example if you wanted to use your own implementation of the `ConnectionSelector` class you can just pass in the *selector_class* parameter.

3.2.1 Transport

class elasticsearch.**Transport**(*hosts*, *connection_class=Urllib3HttpConnection*, *connection_pool_class=ConnectionPool*, *nodes_to_host_callback=construct_hosts_list*, *sniff_on_start=False*, *sniffer_timeout=None*, *sniff_on_connection_fail=False*, *serializer=JSONSerializer()*, *max_retries=3*, *** kwargs*)
Encapsulation of transport-related to logic. Handles instantiation of the individual connections as well as creating a connection pool to hold them.

Main interface is the *perform_request* method.

Parameters

- **hosts** – list of dictionaries, each containing keyword arguments to create a *connection_class* instance
- **connection_class** – subclass of `Connection` to use
- **connection_pool_class** – subclass of `ConnectionPool` to use
- **host_info_callback** – callback responsible for taking the node information from */_cluser/nodes*, along with already extracted information, and producing a list of arguments (same as *hosts* parameter)
- **sniff_on_start** – flag indicating whether to obtain a list of nodes from the cluser at startup time

- **sniffer_timeout** – number of seconds between automatic sniffs
- **sniff_on_connection_fail** – flag controlling if connection failure triggers a sniff
- **sniff_timeout** – timeout used for the sniff request - it should be a fast api call and we are talking potentially to more nodes so we want to fail quickly.
- **serializer** – serializer instance
- **serializers** – optional dict of serializer instances that will be used for deserializing data coming from the server. (key is the mimetype)
- **default_mimetype** – when no mimetype is specified by the server response assume this mimetype, defaults to *'application/json'*
- **max_retries** – maximum number of retries before an exception is propagated
- **send_get_body_as** – for GET requests with body this option allows you to specify an alternate way of execution for environments that don't support passing bodies with GET requests. If you set this to 'POST' a POST method will be used instead, if to 'source' then the body will be serialized and passed as a query parameter *source*.

Any extra keyword arguments will be passed to the *connection_class* when creating and instance unless overriden by that connection's options provided as part of the hosts parameter.

add_connection(*host*)
Create a new `Connection` instance and add it to the pool.

Parameters **host** – kwargs that will be used to create the instance

get_connection()
Retreive a `Connection` instance from the `ConnectionPool` instance.

mark_dead(*connection*)
Mark a connection as dead (failed) in the connection pool. If sniffing on failure is enabled this will initiate the sniffing process.

Parameters **connection** – instance of `Connection` that failed

perform_request(*method*, *url*, *params=None*, *body=None*)
Perform the actual request. Retrieve a connection from the connection pool, pass all the information to it's perform_request method and return the data.

If an exception was raised, mark the connection as failed and retry (up to *max_retries* times).

If the operation was succesful and the connection used was previously marked as dead, mark it as live, resetting it's failure count.

Parameters

- **method** – HTTP method to use
- **url** – absolute url (without host) to target
- **params** – dictionary of query parameters, will be handed over to the underlying `Connection` class for serialization
- **body** – body of the request, will be serializes using serializer and passed to the connection

set_connections(*hosts*)
Instantiate all the connections and crate new connection pool to hold them. Tries to identify unchanged hosts and re-use existing `Connection` instances.

Parameters **hosts** – same as *__init__*

sniff_hosts()
Obtain a list of nodes from the cluster and create a new connection pool using the information retrieved.

To extract the node connection parameters use the *nodes_to_host_callback*.

3.2.2 Connection Pool

class elasticsearch.**ConnectionPool**(*connections*, *dead_timeout=60*, *selector_class=RoundRobinSelector*, *randomize_hosts=True*, *** kwargs*)
Container holding the Connection instances, managing the selection process (via a ConnectionSelector) and dead connections.

It's only interactions are with the Transport class that drives all the actions within *ConnectionPool*.

Initially connections are stored on the class as a list and, along with the connection options, get passed to the *ConnectionSelector* instance for future reference.

Upon each request the *Transport* will ask for a *Connection* via the *get_connection* method. If the connection fails (it's *perform_request* raises a *ConnectionError*) it will be marked as dead (via *mark_dead*) and put on a timeout (if it fails N times in a row the timeout is exponentially longer - the formula is *default_timeout * 2 ** (fail_count - 1)*). When the timeout is over the connection will be resurrected and returned to the live pool. A connection that has been peviously marked as dead and succeedes will be marked as live (it's fail count will be deleted).

Parameters

- **connections** – list of tuples containing the Connection instance and it's options
- **dead_timeout** – number of seconds a connection should be retired for after a failure, increases on consecutive failures
- **timeout_cutoff** – number of consecutive failures after which the timeout doesn't increase
- **selector_class** – ConnectionSelector subclass to use
- **randomize_hosts** – shuffle the list of connections upon arrival to avoid dog piling effect across processes

get_connection()
Return a connection from the pool using the *ConnectionSelector* instance.

It tries to resurrect eligible connections, forces a resurrection when no connections are availible and passes the list of live connections to the selector instance to choose from.

Returns a connection instance and it's current fail count.

mark_dead(*connection*, *now=None*)
Mark the connection as dead (failed). Remove it from the live pool and put it on a timeout.

Parameters **connection** – the failed instance

mark_live(*connection*)
Mark connection as healthy after a resurrection. Resets the fail counter for the connection.

Parameters **connection** – the connection to redeem

resurrect(*force=False*)
Attempt to resurrect a connection from the dead pool. It will try to locate one (not all) eligible (it's timeout is over) connection to return to th live pool.

Parameters **force** – resurrect a connection even if there is none eligible (used when we have no live connections)

3.2.3 Connection Selector

class elasticsearch.**ConnectionSelector**(*opts*)

Simple class used to select a connection from a list of currently live connection instances. In init time it is passed a dictionary containing all the connections' options which it can then use during the selection process. When the *select* method is called it is given a list of *currently* live connections to choose from.

The options dictionary is the one that has been passed to Transport as *hosts* param and the same that is used to construct the Connection object itself. When the Connection was created from information retrieved from the cluster via the sniffing process it will be the dictionary returned by the *host_info_callback*.

Example of where this would be useful is a zone-aware selector that would only select connections from it's own zones and only fall back to other connections where there would be none in it's zones.

Parameters **opts** – dictionary of connection instances and their options

select(*connections*)

Select a connection from the given list.

Parameters **connections** – list of live connections to choose from

3.2.4 Connection

class elasticsearch.**Connection**(*host='localhost'*, *port=9200*, *url_prefix=''*, *timeout=10*, ***kwargs*)

Class responsible for maintaining a connection to an Elasticsearch node. It holds persistent connection pool to it and it's main interface (*perform_request*) is thread-safe.

Also responsible for logging.

Parameters

- **host** – hostname of the node (default: localhost)
- **port** – port to use (default: 9200)
- **url_prefix** – optional url prefix for elasticsearch
- **timeout** – default timeout in seconds (default: 10)

log_request_fail(*method*, *full_url*, *body*, *duration*, *status_code=None*, *exception=None*)

Log an unsuccessful API call.

log_request_success(*method*, *full_url*, *path*, *body*, *status_code*, *response*, *duration*)

Log a successful API call.

3.3 Transport classes

List of transport classes that can be used, simply import your choice and pass it to the constructor of Elasticsearch as *connection_class*. Note that Thrift and Memcached protocols are experimental and require a plugin to be installed in your cluster as well as additional dependencies (*thrift==0.9* and *pylibmc==1.2*).

For example to use the thrift connection just import it and use it. The connection classes are aware of their respective default ports (9500 for thrift) so there is no need to specify them unless modified:

```
from elasticsearch import Elasticsearch, ThriftConnection
es = Elasticsearch(connection_class=ThriftConnection)
```

3.3.1 Connection

class elasticsearch.connection.**Connection**(*host='localhost'*, *port=9200*, *url_prefix=''*, *timeout=10*, ***kwargs*)

Class responsible for maintaining a connection to an Elasticsearch node. It holds persistent connection pool to it and it's main interface (*perform_request*) is thread-safe.

Also responsible for logging.

Parameters

- **host** – hostname of the node (default: localhost)
- **port** – port to use (default: 9200)
- **url_prefix** – optional url prefix for elasticsearch
- **timeout** – default timeout in seconds (default: 10)

3.3.2 Urllib3HttpConnection

class elasticsearch.connection.**Urllib3HttpConnection**(*host='localhost'*, *port=9200*, *http_auth=None*, *use_ssl=False*, *maxsize=10*, ***kwargs*)

Default connection class using the *urllib3* library and the http protocol.

Parameters

- **http_auth** – optional http auth information as either ':' separated string or a tuple
- **use_ssl** – use ssl for the connection if *True*
- **maxsize** – the maximum number of connections which will be kept open to this host.

3.3.3 RequestsHttpConnection

class elasticsearch.connection.**RequestsHttpConnection**(*host='localhost'*, *port=9200*, *http_auth=None*, *use_ssl=False*, ***kwargs*)

Connection using the *requests* library.

Parameters

- **http_auth** – optional http auth information as either ':' separated string or a tuple
- **use_ssl** – use ssl for the connection if *True*

3.3.4 ThriftConnection

class elasticsearch.connection.**ThriftConnection**(*host='localhost'*, *port=9500*, *framed_transport=False*, *use_ssl=False*, ***kwargs*)

Connection using the *thrift* protocol to communicate with elasticsearch.

See https://github.com/elasticsearch/elasticsearch-transport-thrift for additional info.

Parameters **framed_transport** – use *TTransport.TFramedTransport* instead of *TTransport.TBufferedTransport*

3.3.5 MemcachedConnection

class elasticsearch.connection.**MemcachedConnection**(*host='localhost'*, *port=11211*, ***kwargs*)

Client using the *pylibmc* python library to communicate with elasticsearch using the memcached protocol. Requires plugin in the cluster.

See https://github.com/elasticsearch/elasticsearch-transport-memcached for more details.

3.4 Helpers

Collection of simple helper functions that abstract some specifics or the raw API.

elasticsearch.helpers.**streaming_bulk**(*client*, *actions*, *chunk_size=500*, *raise_on_error=False*, *expand_action_callback=<function expand_action at 0x396ab90>*, ***kwargs*)

Streaming bulk consumes actions from the iterable passed in and yields results per action. For non-streaming usecases use `bulk()` which is a wrapper around streaming bulk that returns summary information about the bulk operation once the entire input is consumed and sent.

This function expects the action to be in the format as returned by `search()`, for example:

```
{
    '_index': 'index-name',
    '_type': 'document',
    '_id': 42,
    '_parent': 5,
    '_ttl': '1d',
    '_source': {
        ...
    }
}
```

Alternatively, if *_source* is not present, it will pop all metadata fields from the doc and use the rest as the document data.

Alternative actions (*_op_type* field defaults to *index*) can be sent as well:

```
{
    '_op_type': 'delete',
    '_index': 'index-name',
    '_type': 'document',
    '_id': 42,
}
{
    '_op_type': 'update',
    '_index': 'index-name',
    '_type': 'document',
    '_id': 42,
    'doc': {'question': 'The life, universe and everything.'}
}
```

Parameters

- **client** – instance of `Elasticsearch` to use
- **actions** – iterable containing the actions to be executed

- **chunk_size** – number of docs in one chunk sent to es (default: 500)
- **raise_on_error** – raise *BulkIndexError* containing errors (as *.errors* from the execution of the last chunk)
- **expand_action_callback** – callback executed on each action passed in, should return a tuple containing the action line and the data line (*None* if data line should be omitted).

`elasticsearch.helpers.`**`bulk`**(*client*, *actions*, *stats_only=False*, ***kwargs*)

Helper for the `bulk()` api that provides a more human friendly interface - it consumes an iterator of actions and sends them to elasticsearch in chunks. It returns a tuple with summary information - number of successfully executed actions and either list of errors or number of errors if *stats_only* is set to *True*.

See `streaming_bulk()` for more information and accepted formats.

Parameters

- **client** – instance of `Elasticsearch` to use
- **actions** – iterator containing the actions
- **stats_only** – if *True* only report number of successful/failed operations instead of just number of successful and a list of error responses

Any additional keyword arguments will be passed to `streaming_bulk()` which is used to execute the operation.

`elasticsearch.helpers.`**`scan`**(*client*, *query=None*, *scroll='5m'*, ***kwargs*)

Simple abstraction on top of the `scroll()` api - a simple iterator that yields all hits as returned by underlining scroll requests.

Parameters

- **client** – instance of `Elasticsearch` to use
- **query** – body for the `search()` api
- **scroll** – Specify how long a consistent view of the index should be maintained for scrolled search

Any additional keyword arguments will be passed to the initial `search()` call.

`elasticsearch.helpers.`**`reindex`**(*client*, *source_index*, *target_index*, *target_client=None*, *chunk_size=500*, *scroll='5m'*)

Reindex all documents from one index to another, potentially (if *target_client* is specified) on a different cluster.

Note: This helper doesn't transfer mappings, just the data.

Parameters

- **client** – instance of `Elasticsearch` to use (for read if *target_client* is specified as well)
- **source_index** – index (or list of indices) to read documents from
- **target_index** – name of the index in the target cluster to populate
- **target_client** – optional, is specified will be used for writing (thus enabling reindex between clusters)
- **chunk_size** – number of docs in one chunk sent to es (default: 500)
- **scroll** – Specify how long a consistent view of the index should be maintained for scrolled search

3.5 Changelog

3.5.1 0.4.5 (dev)

- *get_alias* now has *name* as another optional parameter due to issue #4539 in es repo. Note that the order of params have changed so if you are not using keyword arguments this is a breaking change.

3.5.2 0.4.4 (2013-12-23)

- *helpers.bulk_index* renamed to *helpers.bulk* (alias put in place for backwards compatibility, to be removed in future versions)
- Added *helpers.streaming_bulk* to consume an iterator and yield results per operation
- *helpers.bulk* and *helpers.streaming_bulk* are no longer limitted to just index operations.
- unicode body (for *incices.analyze* for example) is now handled correctly
- changed *perform_request* on *Connection* classes to return headers as well. This is a backwards incompatible change for people who have developed their own connection class.
- changed deserialization mechanics. Users who provided their own serializer that didn't extend *JSONSerializer* need to specify a *mimetype* class attribute.
- minor bug fixes

3.5.3 0.4.3 (2013-10-22)

- Fixes to *helpers.bulk_index*, better error handling
- More benevolent *hosts* argument parsing for *Elasticsearch*
- *requests* no longer required (nor recommended) for install

3.5.4 0.4.2 (2013-10-08)

- *ignore* param acceted by all APIs
- Fixes to *helpers.bulk_index*

3.5.5 0.4.1 (2013-09-24)

Initial release.

CHAPTER 4

License

CHAPTER 5

Indices and tables

- *genindex*
- *modindex*
- *search*

e