
mysolr Documentation

Release 0.5.1

Rubén Abad, Miguel Olivares

April 11, 2012

CONTENTS

mysolr was born to be a fast and easy-to-use client for Apache Solr's API and because existing Python clients didn't fulfill these conditions.

Since version 0.5 mysolr supports Python 3 except concurrent search feature.

BASIC USAGE

```
from mysolr import Solr

# Default connection to localhost:8080
solr = Solr()

# All solr params are supported!
query = {'q' : '*:*', 'facet' : 'true', 'facet.field' : 'foo'}
response = solr.search(**query)

# do stuff with documents
for document in response.documents:
    # modify field 'foo'
    document['foo'] = 'bar'

# update index with modified documents
solr.update(response.documents, commit=True)
```


CONTENTS

2.1 Installation

To install mysolr from Pypi:

```
pip install mysolr
```

From source code:

```
python setup.py install
```

2.1.1 Dependencies

Mysolr uses `requests` module for sending HTTP requests. So, if you install mysolr from source code you have to `install` it.

2.2 User Guide

2.2.1 Connecting to Solr

Use `mysolr.Solr` object to connect to a Solr instance.

```
from mysolr import Solr

# Default connection. Connecting to http://localhost:8080/solr/
solr = Solr()

# Custom connection
solr = Solr('http://foo.bar:9090/solr/')
```

2.2.2 Querying to Solr

Making a query to Solr is very easy, just call `search` method with your query.

```
from mysolr import Solr

solr = Solr()
# Search for all documents
```

```
response = solr.search(q='*:*')
# Get documents
documents = response.documents
```

Besides, all available Solr query params are supported. So making a query using pagination would be as simple as

```
from mysolr import Solr

solr = Solr()

# Get 10 documents
response = solr.search(q='*:*', rows=10, start=0)
```

2.2.3 Facets

This is a query example using facets with mysolr.

```
from mysolr import Solr

solr = Solr()
# Search for all documents facets by field foo
query = {'q' : '*:*', 'facet' : 'true', 'facet.field' : 'foo'}
response = solr.search(**query)
# Get documents
documents = response.documents
# Get facets
facets = response.facets
```

Facets are parsed and can be accessed by retrieving `facets` attribute from the `SolrResponse` object. Facets look like this:

```
{
  'facet_dates': {},
  'facet_fields': {'foo': {'value1': 2, 'value2': 2}},
  'facet_queries': {},
  'facet_ranges': {}
}
```

2.2.4 Spellchecker

This is an example of a query that uses the spellcheck component.

```
from mysolr import Solr

solr = Solr()

# Spell check query
query = {
  'q' : 'helo wold',
  'spellcheck' : 'true',
  'spellcheck.collate' : 'true',
  'spellcheck.build' : 'true'
}

response = solr.search(**query)
```

Spellchecker results are parsed and can be accessed by getting the `spellcheck` attribute from the `SolrResponse` object:

```
{'collation': 'Hello world',
 'correctlySpelled': False,
 'suggestions': {
   'helo': {'endOffset': 4,
            'numFound': 1,
            'origFreq': 0,
            'startOffset': 0,
            'suggestion': [{'freq': 14,
                             'word': 'hello'}]},
   'wold': {'endOffset': 9,
            'numFound': 1,
            'origFreq': 0,
            'startOffset': 5,
            'suggestion': [{'freq': 14, 'word': 'world'}]}}
```

2.2.5 Stats

`stats` attribute is just a shortcut to stats result. It is not parsed and has the format sent by Solr.

2.2.6 Highlighting

Like stats, `highlighting` is just a shortcut.

2.2.7 Concurrent searches

As `mysolr` is using requests, it is possible to make concurrent queries thanks to `requests.async`

```
from mysolr import Solr
solr = Solr()
# queries
queries = [
    {
        'q' : '*:*'
    },
    {
        'q' : 'foo:bar'
    }
]

# using 10 threads
responses = solr.async_search(queries, size=10)
```

Using concurrent searches

It's needed `Gevent` module in order to use `requests.async`, so if you need concurrent searches, you must install `Gevent`

2.2.8 Indexing documents

```
from mysolr import Solr

solr = Solr()

# Create documents
documents = [
    {'id' : 1,
     'field1' : 'foo'
    },
    {'id' : 2,
     'field2' : 'bar'
    }
]

# Index using json is faster!
solr.update(documents, 'json', commit=False)

# Manual commit
solr.commit()
```

2.3 Recipes

2.3.1 Solr backup

How to copy all documents from one solr server to another.

```
from mysolr import Solr

PACKET_SIZE = 5000

solr_source = Solr('http://server1:8080/solr/')
solr_target = Solr('http://server2:8080/solr/')

# Get the number of documents of the source index
n_documents = solr_source.search(q='*:*').total_results

for start in range(0, n_documents, PACKET_SIZE):
    resp = solr_source.search(q='*:*', rows=PACKET_SIZE, start=start)
    source_docs = resp.documents
    solr_target.update(source_docs)
```

2.4 Classes

2.4.1 Solr class

```
class mysolr.Solr (base_url='http://localhost:8080/solr/')
    Acts as an easy-to-use interface to Solr.

    async_search (queries, size=10, resource='select')
        Asynchronous search using async module from requests.
```

Parameters

- **queries** – List of queries. Each query is a dictionary containing any of the available Solr query parameters described in <http://wiki.apache.org/solr/CommonQueryParameters>. ‘q’ is a mandatory parameter.
- **size** – Size of threadpool
- **resource** – Request dispatcher. ‘select’ by default.

commit (*wait_flush=True, wait_searcher=True, expunge_deletes=False*)
Sends a commit message to Solr.

Parameters

- **wait_flush** – Block until index changes are flushed to disk (default is True).
- **wait_searcher** – Block until a new searcher is opened and registered as the main query searcher, making the changes visible (default is True).
- **expunge_deletes** – Merge segments with deletes away (default is False)

delete_by_key (*identifier, commit=True*)
Sends an ID delete message to Solr.

Parameters commit – If True, sends a commit message after the operation is executed.

delete_by_query (*query, commit=True*)
Sends a query delete message to Solr.

Parameters commit – If True, sends a commit message after the operation is executed.

optimize (*wait_flush=True, wait_searcher=True, max_segments=1*)
Sends an optimize message to Solr.

Parameters

- **wait_flush** – Block until index changes are flushed to disk (default is True)
- **wait_searcher** – Block until a new searcher is opened and registered as the main query searcher, making the changes visible (default is True)
- **max_segments** – Optimizes down to at most this number of segments (default is 1)

rollback ()
Sends a rollback message to Solr server.

search (*resource='select', **kwargs*)
Queries Solr with the given kwargs and returns a SolrResponse object.

Parameters

- **resource** – Request dispatcher. ‘select’ by default.
- ****kwargs** – Dictionary containing any of the available Solr query parameters described in <http://wiki.apache.org/solr/CommonQueryParameters>. ‘q’ is a mandatory parameter.

update (*documents, input_type='json', commit=True*)
Sends an update/add message to add the array of hashes(documents) to Solr.

Parameters

- **documents** – A list of solr-compatible documents to index. You should use unicode strings for text/string fields.
- **input_type** – The format which documents are sent. Remember that json is not supported until version 3.
- **commit** – If True, sends a commit message after the operation is executed.

2.4.2 SolrResponse class

```
class mysolr.SolrResponse(solr_response)
    Parse solr response and make it accesible.

    documents = None
        Documents list.

    facets = None
        Facets parsed as a dict.

    highlighting = None
        Shortcut to highlighting result

    qtime = None
        Query time.

    raw_response = None
        Solr full response.

    spellcheck = None
        Spellcheck result parsed into a more readable object.

    start = None
        Offset.

    stats = None
        Shortcut to stats results

    status = None
        Response status.

    total_results = None
        Number of results.

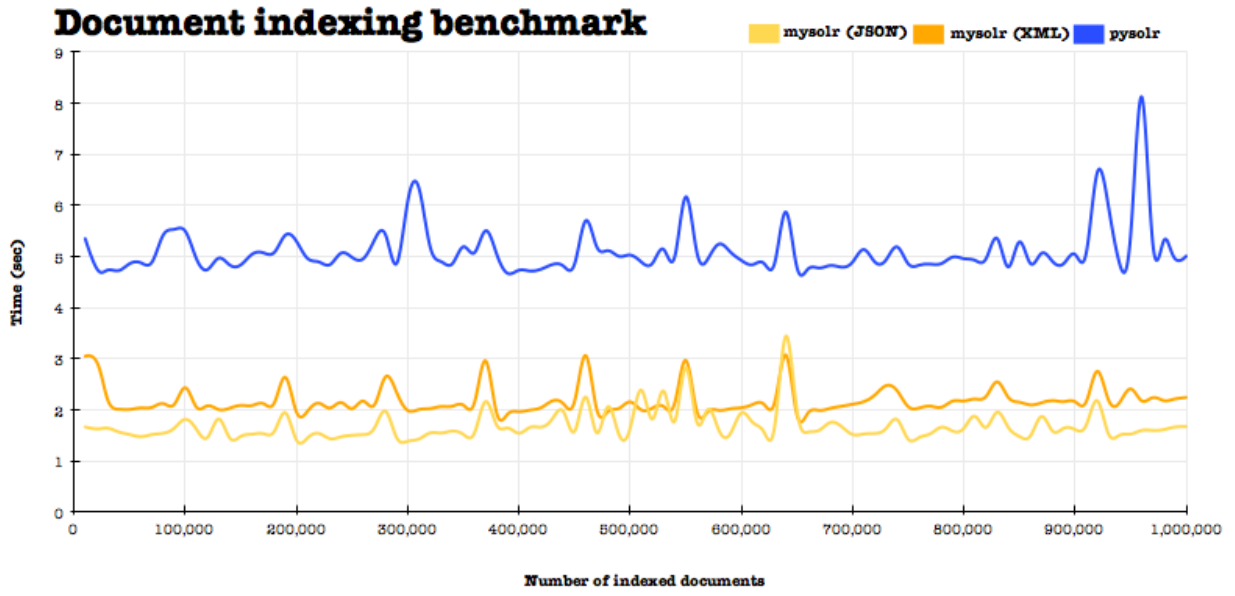
    url = None
        Solr query URL
```

2.5 Benchmark

One of the main goals of mysolr is to be the fastest python client of Solr. In this section you can see the performance of mysolr in different situations.

2.5.1 Indexing

The picture below is a comparison between mysolr and other clients at indexing time.



REFERENCES

We would like to thank the following developers their work and inspiration:

- The Apache Solr ['s committers](#)
- Kenneth Reitz, [Requests](#) creator

RELATED PROJECTS

Other Python projects Apache Solr related:

- [solrpy](#)
- [pysolr](#)
- [djangosolr](#)

PYTHON MODULE INDEX

m

`mysolr, ??`