“Cool, Bonsai, Cool”

An introduction to

Clinton Gormley, YAPC::EU 2011

Why do I need a search engine?

Top: Computers: Programming: Languages: Perl *(789)*

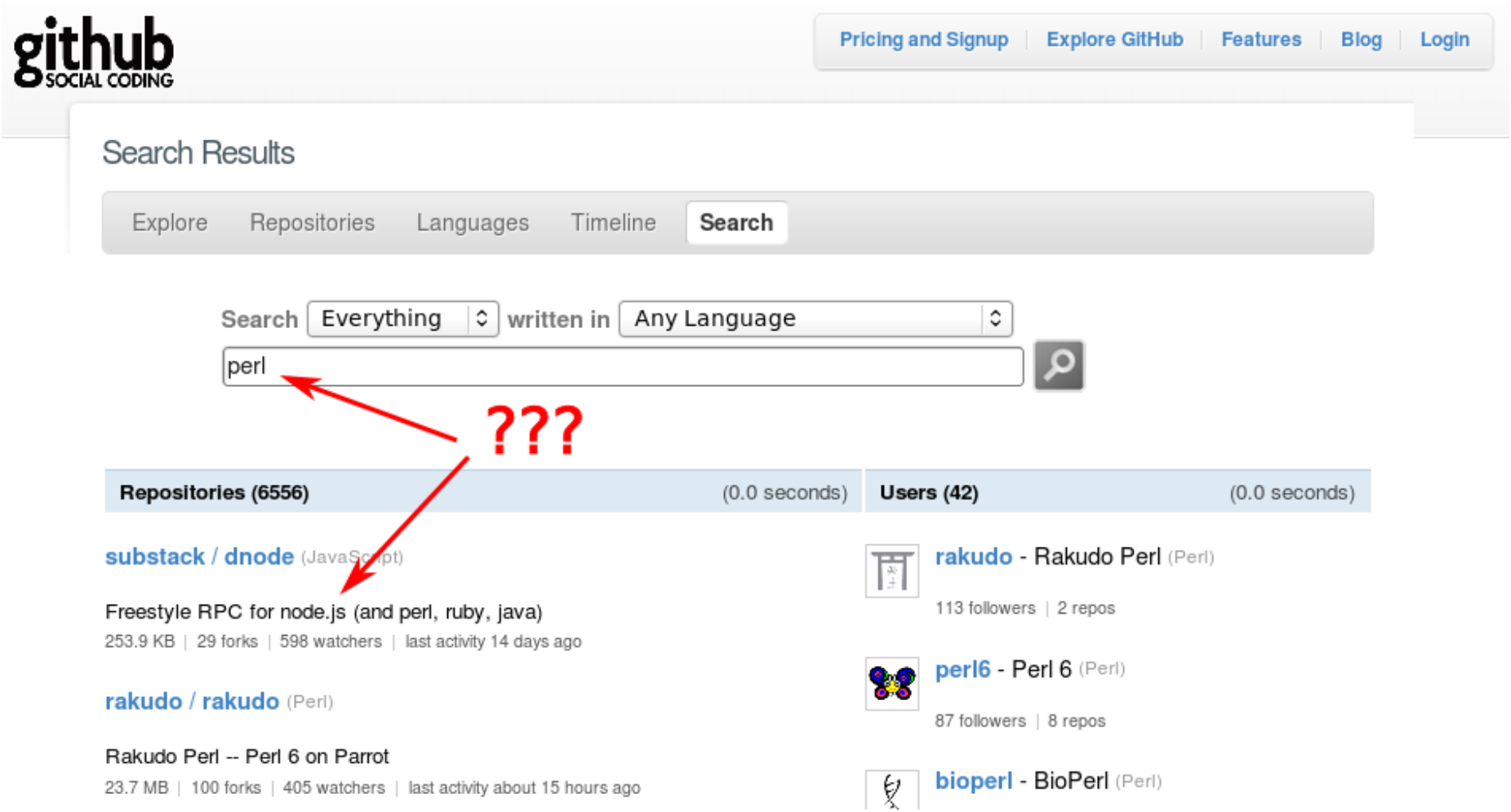
Description

- Wall, Larry *(7)*

- Chats and Forums *(4)*
- Commercial Services *(38)*
- Conferences *(5)*
- Directories *(5)*
- Documentation *(19)*
- FAQs, Help, and Tutorials *(63)*
- Magazines and E-zines *(4)*
- Modules *(144)*
- Personal Pages *(40)*
- Poetry *(2)*
- Scripts *(18)*
- Tools *(21)*
- User Groups *(92)*


2000

Search is how we find stuff


github
SOCIAL CODING
Pricing and Signup
Explore GitHub
Features
Blog
Login
Search Results
Explore
Repositories
Languages
Timeline
Search
Search
Everything
written in
Any Language
perl
???
Repositories (6556)
(0.0 seconds)
substack / dnode (JavaScript)
Freestyle RPC for node.js (and perl, ruby, java)
253.9 KB | 29 forks | 598 watchers | last activity 14 days ago
rakudo / rakudo (Perl)
Rakudo Perl -- Perl 6 on Parrot
23.7 MB | 100 forks | 405 watchers | last activity about 15 hours ago
Users (42)
(0.0 seconds)
rakudo - Rakudo Perl (Perl)
113 followers | 2 repos
perl6 - Perl 6 (Perl)
87 followers | 8 repos
bioperl - BioPerl (Perl)

Google

site:github.com perl

About 22,400 results (0.20 seconds)

- Everything
- Images
- Videos
- News
- Shopping
- Blogs
- Books
- More

Perl - GitHub
https://github.com/languages/**Perl** - Cached
Perl is the #8 most popular language on GitHub. Explore · Repositories · Languages · Timeline · Search · **Perl** · Recently Created · Recently Updated ...

log4perl - log4j for **Perl**
mschilli.github.com/log4**perl**/ - Cached
Log::Log4perl is a **Perl** port of the widely popular log4j logging package. Logging beats a debugger if you want to know what's going on in your code during ...

ericblue/**Perl**-FitBit-API - GitHub
https://github.com/ericblue/**Perl**-FitBit-API - Cached
Perl-FitBit-API - Provides an OO API for fetching fitness data from fitbit.com. Currently there is no official API, however data is retrieved using XML ...

shlomif/**perl** - GitHub
https://github.com/shlomif/**perl** - Cached
perl mirror. ... shlomif / **perl** forked from mirrors/**perl** · Admin · Watch Unwatch · Fork; Your Fork. 2 · 78 · Source · Commits · Network · Pull Requests (0) ...

marcgreen/**perl** - GitHub
https://github.com/marcgreen/**perl** - Cached
The **Perl** programming language — Read more .... There are also many **Perl** books available, covering a wide variety of topics, from various publishers. ...

How does a search engine work?


Magic!

Acme::**Magic**8Ball
Acme::**Magic**::Pony
Config::**Magic**
File::**Magic**
File::MimeInfo::**Magic**
File::M**Magic**::XS
MagicTemplate
Meta::File::M**Magic**
MRO::**Magic**
Template::**Magic**
Template::**Magic**::Pager
Test::**Magic**
XS::**Magic**Ext
XS::Object::**Magic**

Magic

==

inverted index

+

relevance scoring

Take some text

```
          Acme::Magic8Ball
          Acme::Magic::Pony
        Config::Magic
          File::Magic
File::MimeInfo::Magic
         File::MMagic::XS
                MagicTemplate
   Meta::File::MMagic
           MRO::Magic
      Template::Magic
      Template::Magic::Pager
          Test::Magic
            XS::MagicExt
    XS::Object::Magic
```

Tokenise it

```
          Acme::Magic8Ball
          Acme::Magic::Pony
        Config::Magic
          File::Magic
File::MimeInfo::Magic
         File::MMagic::XS
                MagicTemplate
   Meta::File::MMagic
           MRO::Magic
      Template::Magic
      Template::Magic::Pager
          Test::Magic
            XS::MagicExt
    XS::Object::Magic
```

Tokenise it

```
          acme  magic 8 ball
          acme  magic pony
        config  magic
          file  magic
file mime info  magic
         file m magic xs
                magic template
   meta  file m magic
           mro  magic
      template  magic
      template  magic pager
          test  magic
            xs  magic ext
    xs  object  magic
```

Find unique tokens/terms

```
          acme  magic 8 ball
          acme  magic pony
        config  magic
          file  magic
file mime info  magic
         file m magic xs
                magic template
   meta  file m magic
           mro  magic
      template  magic
      template  magic pager
          test  magic
            xs  magic ext
    xs  object  magic
```

Find unique tokens/terms

8
acme
ball
config
ext
file
info
m
magic
meta
mime
mro
object
pager
pony
template
test
xs

Map terms to documents

	acme	file	magic	mime	template	xs
Acme::Magic8Ball	■		■			
Acme::Magic::Pony	■		■			
File::Magic		■	■			
File::MimeInfo::Magic		■	■	■		
MagicTemplate			■		■	
Template::Magic			■		■	
Template::Magic::Pager			■		■	
XS::Object::Magic			■			■
XS::MagicExt			■			■
File::MMagic::XS		■	■			■

Search for: “`file xs`”

	acme	file	magic	mime	template	xs
Acme::Magic8Ball	■		■			
Acme::Magic::Pony	■		■			
File::Magic		■	■			
File::MimeInfo::Magic		■	■	■		
MagicTemplate			■		■	
Template::Magic			■		■	
Template::Magic::Pager			■		■	
XS::Object::Magic			■			■
XS::MagicExt			■			■
File::MMagic::XS		■	■			■

Search for: “`file xs`”

	acme	file	magic	mime	template	xs
Acme::Magic8Ball						
Acme::Magic::Pony						
File::Magic						
File::MimeInfo::Magic						
MagicTemplate						
Template::Magic						
Template::Magic::Pager						
XS::Object::Magic						
XS::MagicExt						
File::MMagic::XS						

But,
not just about finding

Sort by

RELEVANCE

Relevance:

How many matching terms does this document contain?

Relevance:

How often does each term appear in **this** document, as a % of its length?

Relevance:

How frequently does each term appear in **all your documents**?

Relevance:

Can be customised

Relevance:

Can be customised
By document or field

Relevance:

Can be customised
By document or field
At index or search time

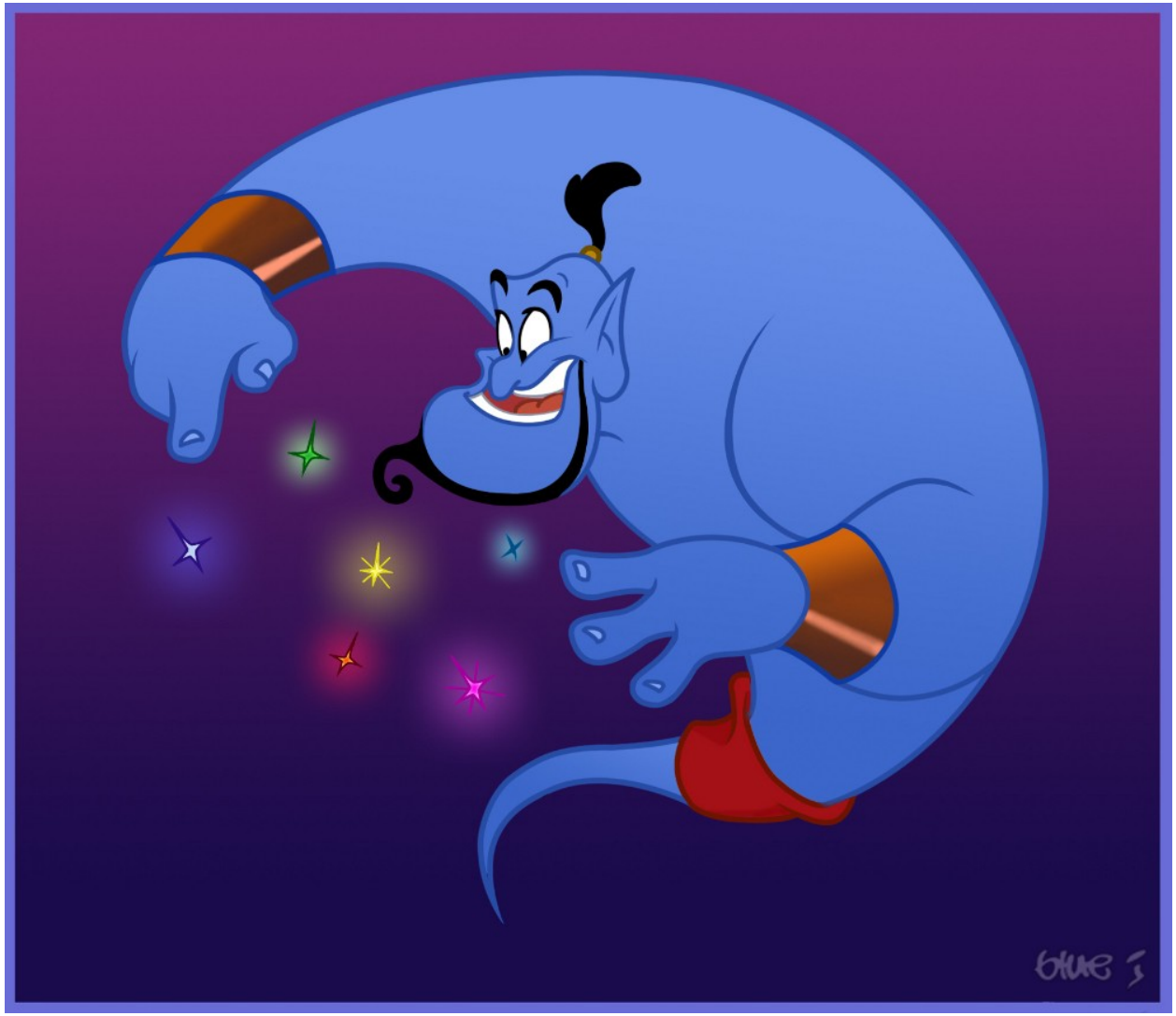
Simple as:

Can be customised
By document or field
At index or search time

FAST!

POWERFUL!

MAGIC!


blue

www.elasticsearch.org

elasticsearch is:

elasticsearch is:

- an Open Source (Apache 2)

elasticsearch is:

- an Open Source (Apache 2)
- distributed

elasticsearch is:

- an Open Source (Apache 2)
- distributed
- RESTful

elasticsearch is:

- an Open Source (Apache 2)
- distributed
- RESTful
- search engine

elasticsearch is:

- an Open Source (Apache 2)
- distributed
- RESTful
- search engine
- built on top of Lucene

Installing elasticsearch:

Latest version at:

`http://www.elasticsearch.org/download/`

```
wget https://github.com/.../elasticsearch-0.17.6.tar.gz
tar -xzf elasticsearch-0.17.6.tar.gz
cd elasticsearch-0.17.6/
./bin/elasticsearch
```

Installing ElasticSearch.pm:

Latest version at:

`https://metacpan.org/module/ElasticSearch`

```
cpanm ElasticSearch
perl -de 0
> use ElasticSearch;
> $e = ElasticSearch->new( trace_calls => 1)
> $e->cluster_health
```

Some terminology

Relational DB	elasticsearch

Some terminology

Relational DB		elasticsearch
database	⇒	index

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type
row	⇒	document

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type
row	⇒	document
column	⇒	field

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type
row	⇒	document
column	⇒	field
schema	⇒	mapping

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type
row	⇒	document
column	⇒	field
schema	⇒	mapping
index	⇒	everything is indexed

Some terminology

Relational DB		elasticsearch
database	⇒	index
table	⇒	type
row	⇒	document
column	⇒	field
schema	⇒	mapping
index	⇒	everything is indexed
SQL	⇒	query DSL

Clustering

Clustering

auto-discovery

Clustering

single master
auto-elected

Clustering

immediate failover
master re-election

Clustering

index
==

Clustering

index
==
1 or more primary shards

Clustering

index
==
1 or more primary shards
+
0 or more replica shards

Clustering

more primary shards

Clustering

more primary shards ⇒ faster indexing

⇒ more scale

Clustering

more primary shards ⇒ faster indexing

⇒ more scale

more replicas

Clustering

more primary shards ⇒ faster indexing

⇒ more scale

more replicas ⇒ faster searching

⇒ more failover

Clustering

Big subject...

http://www.elasticsearch.org/videos/2011/08/09/road-to-a-distributed-searchengine-berlinbuzzwords.html

http://berlinbuzzwords.de/sites/berlinbuzzwords.de/files/elasticsearch-bbuzz2011.pdf

Document oriented:

Document oriented:

No ORM required

Document oriented:

JSON in ⇔ JSON out

Schema free

Dynamic mapping

Schema free

Dynamic (or strict) mapping

Unknown field?

elasticsearch
guesses the type

elasticsearch
guesses the type
and indexes it

Put data in:

```
$e->index(

);
```

Put data in:

```
$e->index(
    index  => 'twitter',

);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',



);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,

);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1, # optional









);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1, # ES always returns the ID









);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,
    data   => {









    }
);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,
    data   => {
        tweet => “ElasticSearch is cool”,





    }
);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,
    data   => {
        tweet => “ElasticSearch is cool”,
        sent  => “2011-08-16 15:15:00”,





    }
);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,
    data   => {
        tweet => “ElasticSearch is cool”,
        sent  => “2011-08-16 15:15:00”,
        user  => {
            name    => “Clinton”,
            user_id => 123
        },

    }
);
```

Put data in:

```
$e->index(
    index  => 'twitter',
    type   => 'tweet',
    id     => 1,
    data   => {
        tweet => “ElasticSearch is cool”,
        sent  => “2011-08-16 15:15:00”,
        user  => {
            name    => “Clinton”,
            user_id => 123
        },
        tags  => [“search”,”perl”],
    }
);
```

Realtime GET

Retrieve your doc immediately

Persistent

No commit required

Get data out:

```
$e->get( index  => 'twitter',
         type   => 'tweet',
         id     => 1);
```

Get data out:

```
$e->get( index  => 'twitter',
         type   => 'tweet',
         id     => 1);
```

```
{
    _index      => 'twitter',
    _type       => 'tweet',
    _id         => 1,









}
```

Get data out:

```
$e->get( index  => 'twitter',
         type   => 'tweet',
         id     => 1);
```

```
{
    _index      => 'twitter',
    _type       => 'tweet',
    _id         => 1,
    _version    => 1,









}
```

Get data out:

```
$e->get( index  => 'twitter',
         type   => 'tweet',
         id     => 1);
```

```
{
    _index      => 'twitter',
    _type       => 'tweet',
    _id         => 1,
    _version    => 1,
    _source     => {
        tweet => “ElasticSearch is cool”,
        sent  => “2011-08-16 15:15:00”,
        user  => {
            name    => “Clinton”,
            user_id => 123
        },
        tags  => ['search','perl'],
    }
}
```

bulk-indexing

bulk-indexing
multi-get

bulk-indexing
multi-get
avoids http latency

bulk-indexing
multi-get
avoids http latency
10x as fast!

Versioning

Versioning

“Optimistic currency control”

Versioning

“Put if absent”

Versioning

Optional

Versioning

Can use external version numbers

So far, all we have is
a NoSQL document store
which is
fast, reliable, scalable & easy to use

So far have is
a NoSQ nt store

fast, reliable, & easy to use

Simple search

```
$e->search(
    index  => 'twitter',
    type   => 'tweet',





);
```

Simple search

```
$e->search(
    index  => ['twitter','facebook'],
    type   => ['tweet','post'],





);
```

Simple search

```
$e->search(
    # all indices
    # all types




);
```

Simple search

```
$e->search(
    index  => 'twitter',
    type   => 'tweet',
    query  => {



    }
);
```

Simple search

```
$e->search(
    index  => 'twitter',
    type   => 'tweet',
    query  => {
        text => {
            _all => 'clinton'
        }
    }
);
```

Simple search

```
$e->search(
    index  => 'twitter',
    type   => 'tweet',
    querybb => 'clinton'




);
```

Simple search

```
$e->search(
    index  => 'twitter',
    type   => 'tweet',
    queryb => 'clinton'

    # ElasticSearch::SearchBuilder,
    # like SQL::Abstract

);
```

Search results

```
{
    took => 1,
    hits => {
        total     => 1,
        max_score => 1,
        hits      => [{
            _score  => 1,
            _index  => 'twitter',
            _type   => 'tweet',
            _id     => 1,
            _source => {
                tweet => “ElasticSearch is cool”,
                sent  => “2011-08-16 15:15:00”,
                user  => {
                    name    => “Clinton”,
                    user_id => 123

                },
                tags  => ['search','perl'],

            }
        }],
    },
    ... other information ...
}
```

Search results

```
{
    took => 1,  # milliseconds
    hits => {
        total     => 1,
        max_score => 1,
        hits      => [{
            _score  => 1,
            _index  => 'twitter',
            _type   => 'tweet',
            _id     => 1,
            _source => {
                tweet => “ElasticSearch is cool”,
                sent  => “2011-08-16 15:15:00”,
                user  => {
                    name    => “Clinton”,
                    user_id => 123

                },
                tags  => ['search','perl'],

            }
        }],
    },
    ... other information ...
}
```

Search results

```
{
    took => 1,
    hits => {
        total     => 1,  # total results
        max_score => 1,
        hits      => [{
            _score  => 1,
            _index  => 'twitter',
            _type   => 'tweet',
            _id     => 1,
            _source => {
                tweet => “ElasticSearch is cool”,
                sent  => “2011-08-16 15:15:00”,
                user  => {
                    name    => “Clinton”,
                    user_id => 123

                },
                tags  => ['search','perl'],

            }
        }],
    },
    ... other information ...
}
```

Search results

```
{
    took => 1,
    hits => {
        total     => 1,
        max_score => 1,
        hits      => [{
            _score  => 1,
            _index  => 'twitter',
            _type   => 'tweet',
            _id     => 1,
            _source => {
                tweet => “ElasticSearch is cool”,
                sent  => “2011-08-16 15:15:00”,
                user  => {
                    name    => “Clinton”,
                    user_id => 123

                },
                tags  => ['search','perl'],

            }
        }],
    },
    ... other information ...
}
```

Search results

```
{
    took => 1,
    hits => {
        total     => 1,
        max_score => 1,
        hits      => [{
            _score  => 1,
            _index  => 'twitter',
            _type   => 'tweet',
            _id     => 1,
            _source => {
                tweet => “ElasticSearch is cool”,
                sent  => “2011-08-16 15:15:00”,
                user  => {
                    name    => “Clinton”,
                    user_id => 123

                },
                tags  => ['search','perl'],

            }
        }],
    },
    ... other information ...
}
```

JSON doc included in results

No need to fetch from DB

Docs visible to search in
near-real time (< 1 second)

`refresh_index()` to force

What can you do with search?

standard text search

Issue search:

User [ ] as Any

Keywords custom script

All (128) | Open (24) | Closed (104)

closed #226 **Script Filter: Support providing a custom script as a filter** v0.09.0 feature

by kimchy - 15 Jun 2010 - closed 15 Jun 2010 1 comment

e new script support (as used in custom _score and script fields). For example: "filtered" : { "query" : { ... }, "filter" : { " script " : { ... " script " : "doc['num1'].value > 1" } } } Custom parameters can also be provided for it: "filtered" : { "query" : { ... }, " ... " script " : { " script " : "doc['num1'].value > param1" "params" : { "param1" : 5 } } } }

closed #1204 **Query DSL: custom _filters_score allow to associate boost on filter instead of script** enhancement v0.17.3 v0.18.0

by kimchy - 3 Aug 2011 - closed 3 Aug 2011

in a custom _filters_score which will improve perf compared to a script .

...with highlighting

Issue search:

User [ ] as Any

Keywords custom script

All (128) | Open (24) | Closed (104)

closed #226 **Script Filter: Support providing a custom script as a filter** v0.09.0 feature

by kimchy - 15 Jun 2010 - closed 15 Jun 2010 1 comment

e new script support (as used in custom _score and script fields). For example: "filtered" : { "query" : { ... }, "filter" : { " script " : { ... " script " : "doc['num1'].value > 1" } } } Custom parameters can also be provided for it: "filtered" : { "query" : { ... }, " ... " script " : { " script " : "doc['num1'].value > param1" "params" : { "param1" : 5 } } } }

closed #1204 **Query DSL: custom _filters_score allow to associate boost on filter instead of script** enhancement v0.17.3 v0.18.0

by kimchy - 3 Aug 2011 - closed 3 Aug 2011

in a custom _filters_score which will improve perf compared to a script .

stemming

Issue search:

User

Keywords flushed

All (32) Open (1) Closed (31)

closed #1180 When flushing, old transaction log is not removed

by kimchy - 29 Jul 2011 - closed 29 Jul 2011

When **flushing**, old transaction log is not removed

closed #187 Optimize API: Change flush and refresh to default to true and not false

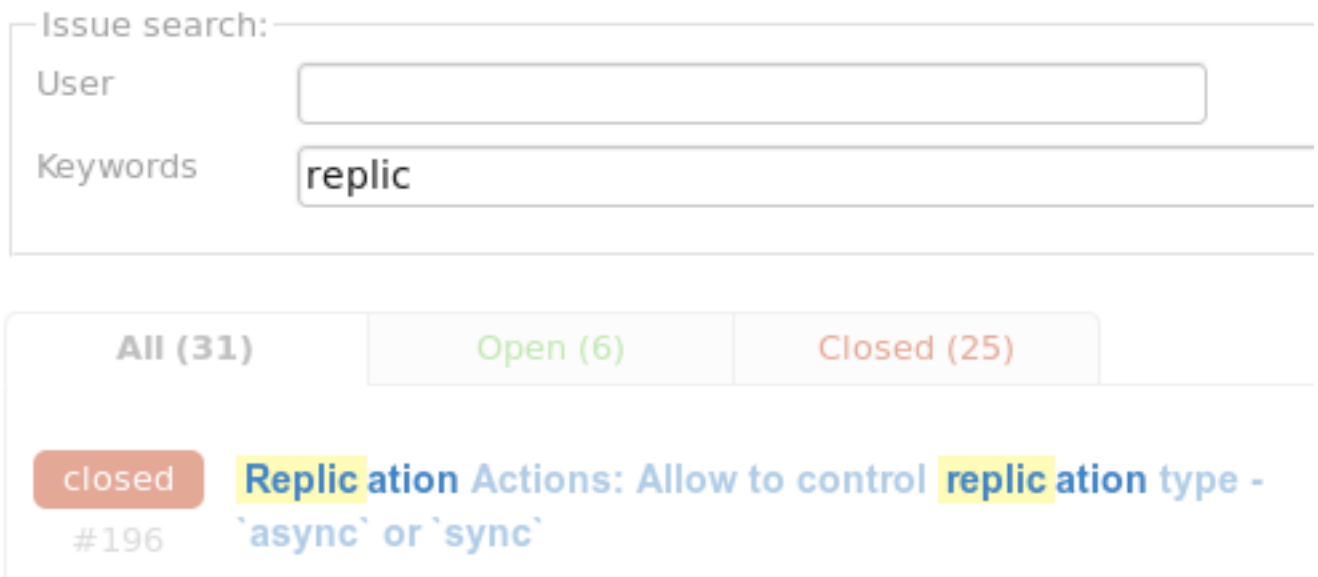
by kimchy - 21 May 2010 - closed 21 May 2010

do a **flush** and refresh after optimize by default then not.

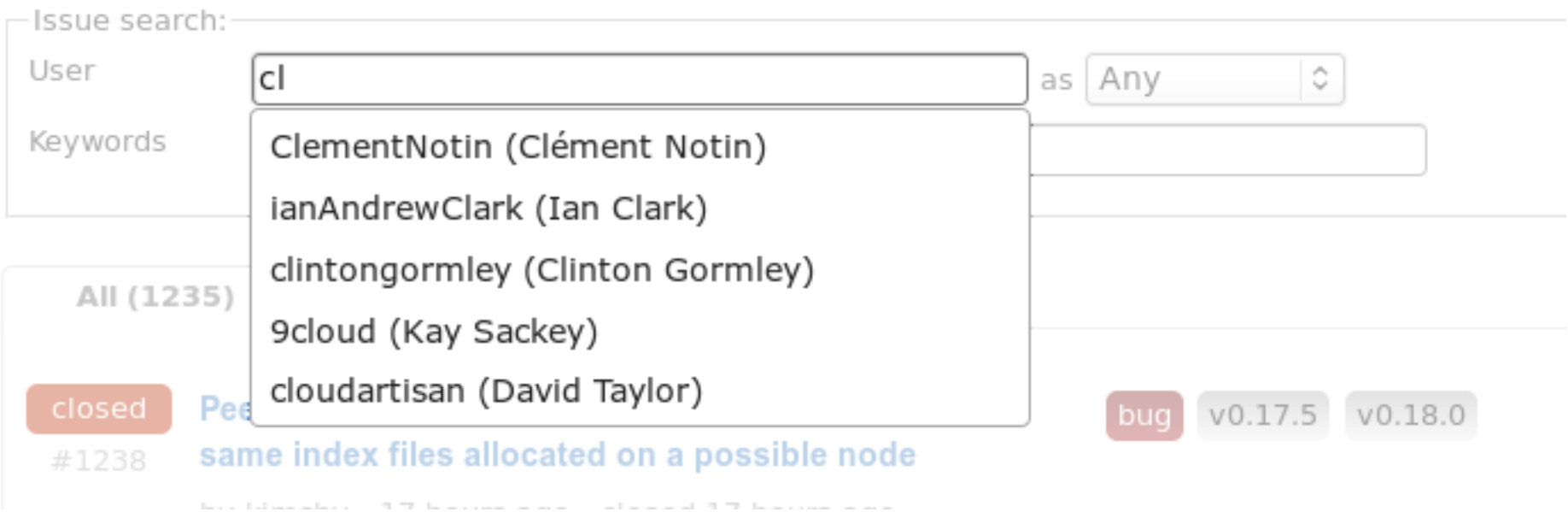
stemming

arabic, armenian, basque, brazilian, bulgarian, catalan, chinese, cjk, czech, danish, dutch, english, finnish, french, galician, german, german2, greek, hindi, hungarian, indonesian, italian, kp, light_finish, light_french, light_german, light_hungarian, light_italian, light_portuguese, light_russian, light_spanish, light_swedish., lovins, minimal_english, minimal_french, minimal_german, minimal_portuguese, norwegian, persian, porter, porter2, portuguese, possessive_english, romanian, russian, spanish, swedish, thai, turkish

ngrams & edge-ngrams



auto-complete



camelCase

Issue search:

User as Any

Keywords deletebyquery

All (5) Open (1) Closed (4)

open DeleteByQuery does not delete when multiple indices and types given

camelCase

Issue search:

User | as Any

Keywords | delete by query

All (514) | Open (97) | Closed (417)

open DeleteByQuery does not delete when multiple indices and types given

camelCase

Issue search:

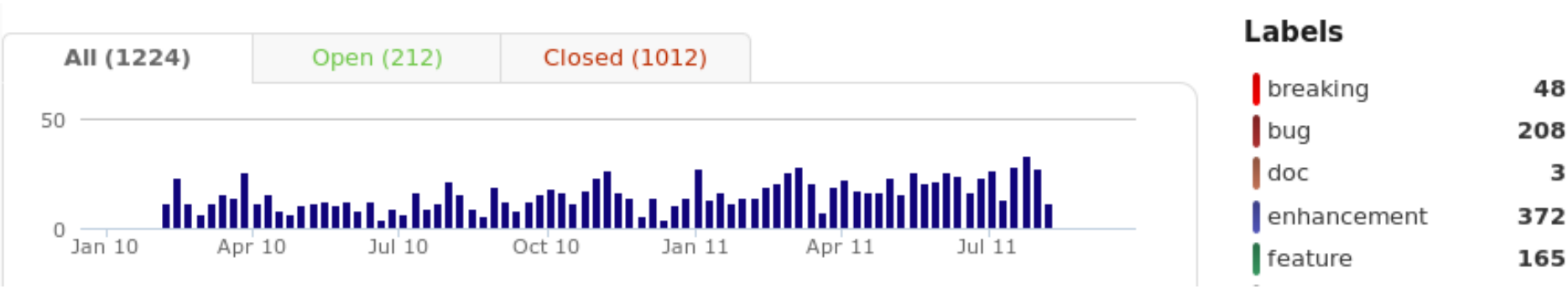
User | as | Any

Keywords | query delete

All (368) | Open (69) | Closed (299)

open | Delete By Query does not delete when multiple indices and types given

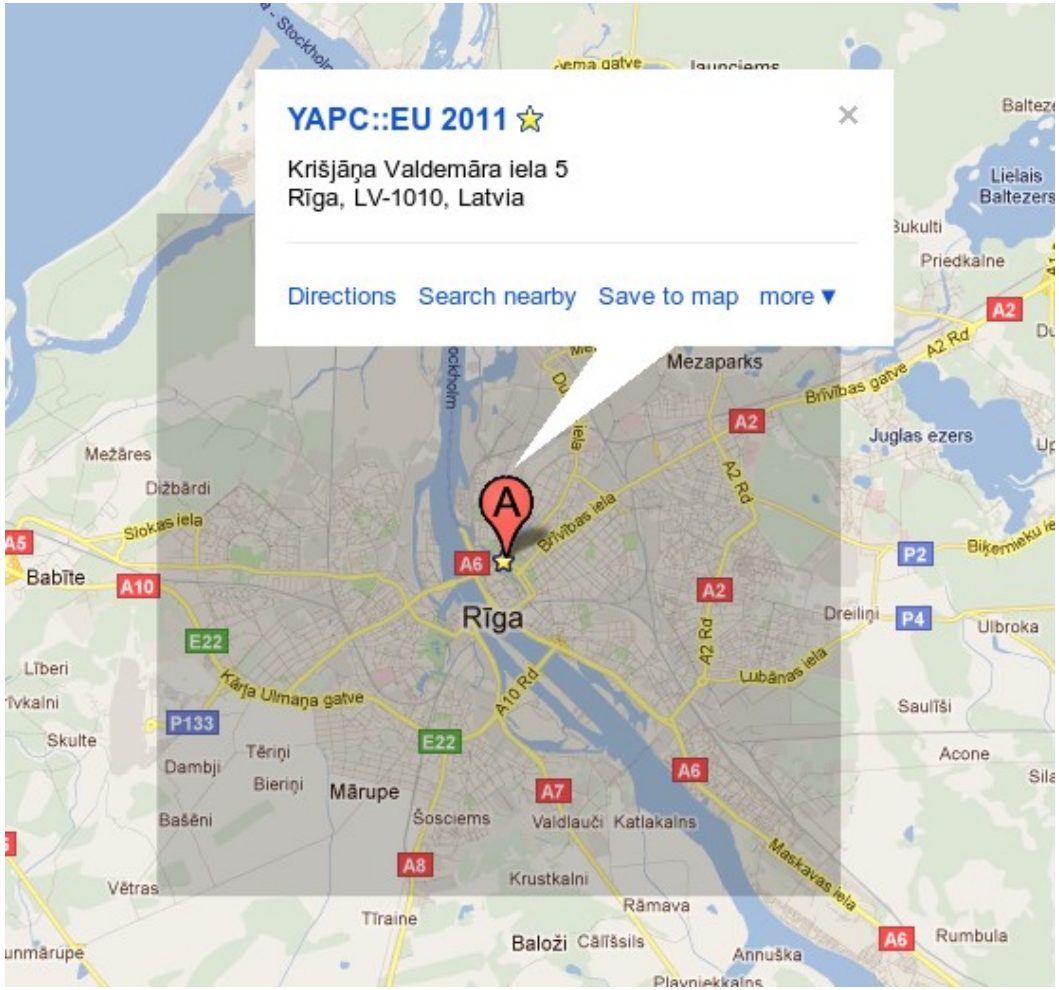
term facets, date histograms



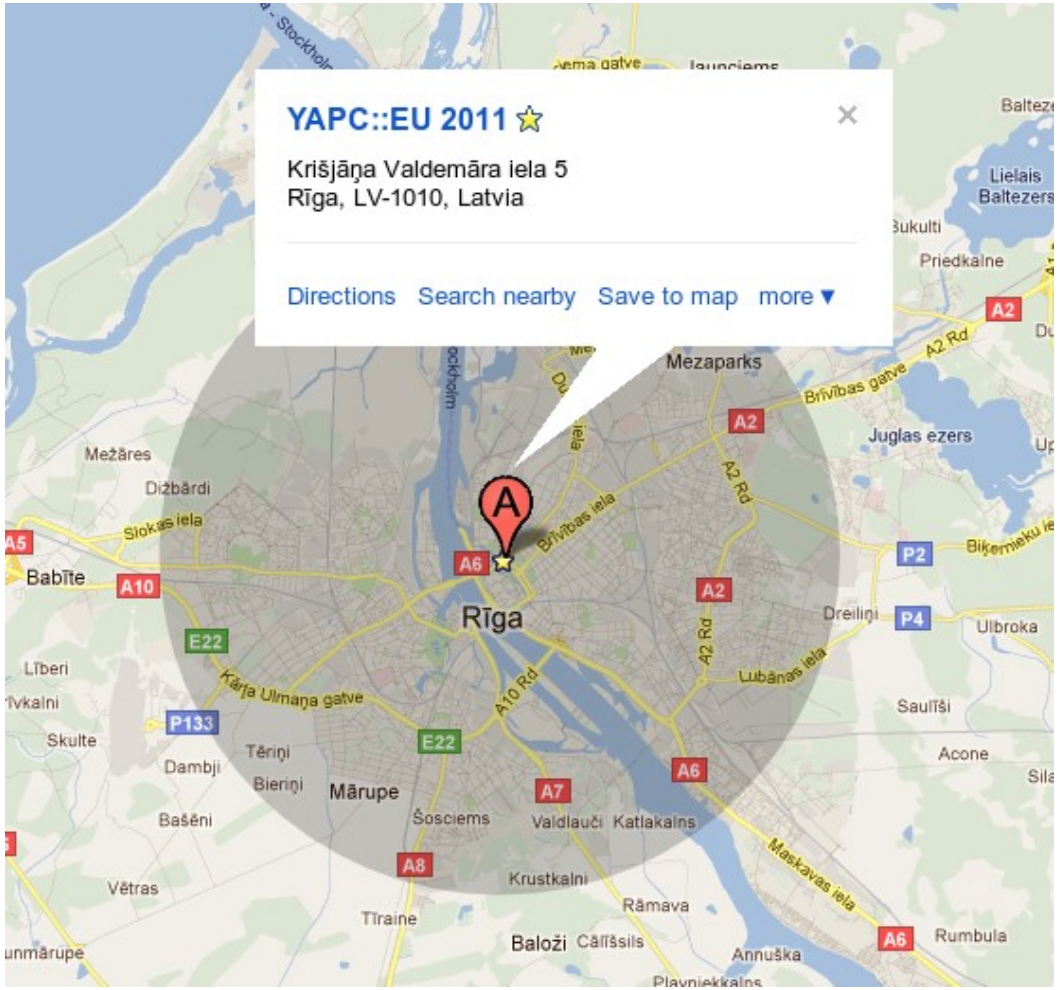
ranges


All (1224)
Open (212)
Closed (1012)
20
0
Jan 10
Apr 10
Jul 10
Oct 10
Jan 11
Apr 11
Jul 11
open
#523
Support for expiring out documents / TTL
by gwittelpp - 16 Nov 2010 - last updated 27 Jul 2011
2 comments
It would be really nice if there was a way to automatically expire out old documents from a particular index. In the simplest form it would be a TTL per index. Once a document's age (based on insert time or maybe another configurable field),...

geo bounding box

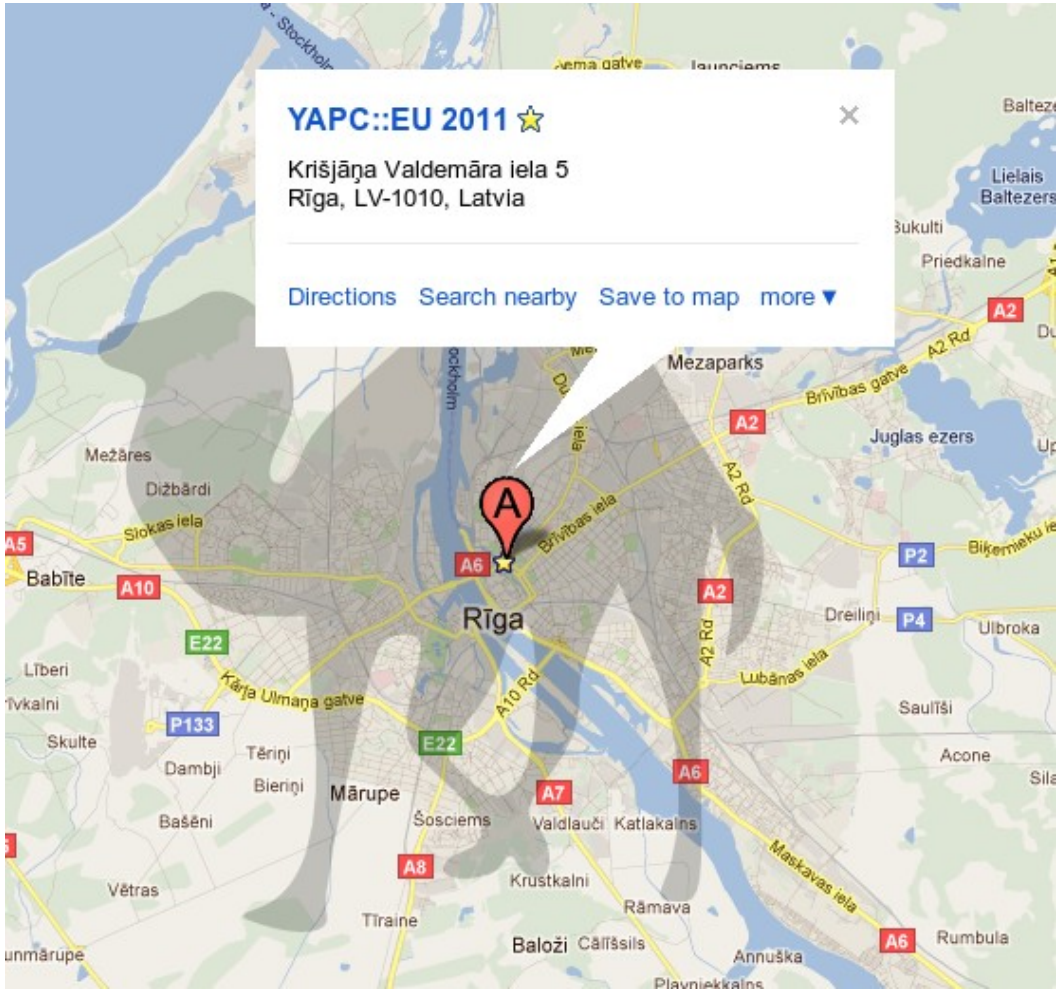


geo distance



geo distance ranges

geo polygons



“Terms of endearment”

The ElasticSearch query language explained

Thurs. 14:35 - Auditorija 301