Welcome To
ElasticSearch
China Conference #1

2013-01-20 Beijing,China

INFINITBYTE

Agenda

1.00 PM： 签到
1.20 PM： 相互介绍
1.50 PM： elasticsearch入门及实例讲解
3.00 PM： 休息
3.10 PM： 自由交流
3.30 PM： elasticsearch架构设计与调优
4.30 PM： 自由交流
6.00 PM： 活动结束

About me

- Medcl
- m@medcl.net
- http://log.medcl.net
- http://github.com/medcl
- http://t.sina.com/medcl
- 2010年接触elasticsearch，版本0.5.1

ElasticSearch Training

Brief Tutorial

Medcl,2013.1.20

你将学到什么？

- 介绍及安装
- 如何索引数据
- 如何构造查询
- Mapping介绍
- 常用 HTTP API 介绍
- 实战指导
- 常用JAVA API介绍
- ElasticSearch.NET介绍

What's elasticsearch

- Full Text Search Engine
- Lucene Based，Written in Java
- "Distributed, (Near) Real Time, Search Engine"
- RESTful，JSON，HTTP，Easy To Debug
- Free Schema（Dynamic Mapping）
- MultiTenant
- Scalable，From One Node To One Thousand Node
- High Availability
- Rich Search Features
- Good Extensibility
- Open Source（Apache 2.0）
- Written by Shay Bannon（Kimchy）（Compass，Another SEF）

安装

- 零配置，开箱即用
- 没有繁琐的安装配置
- 各版本下载地址：
  http://www.elasticsearch.org/download

```
wget
http://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-0.20.2.zip
unzip elasticsearch-0.20.2.zip
cd  elasticsearch-0.20.2/bin
./elasticsearch
```

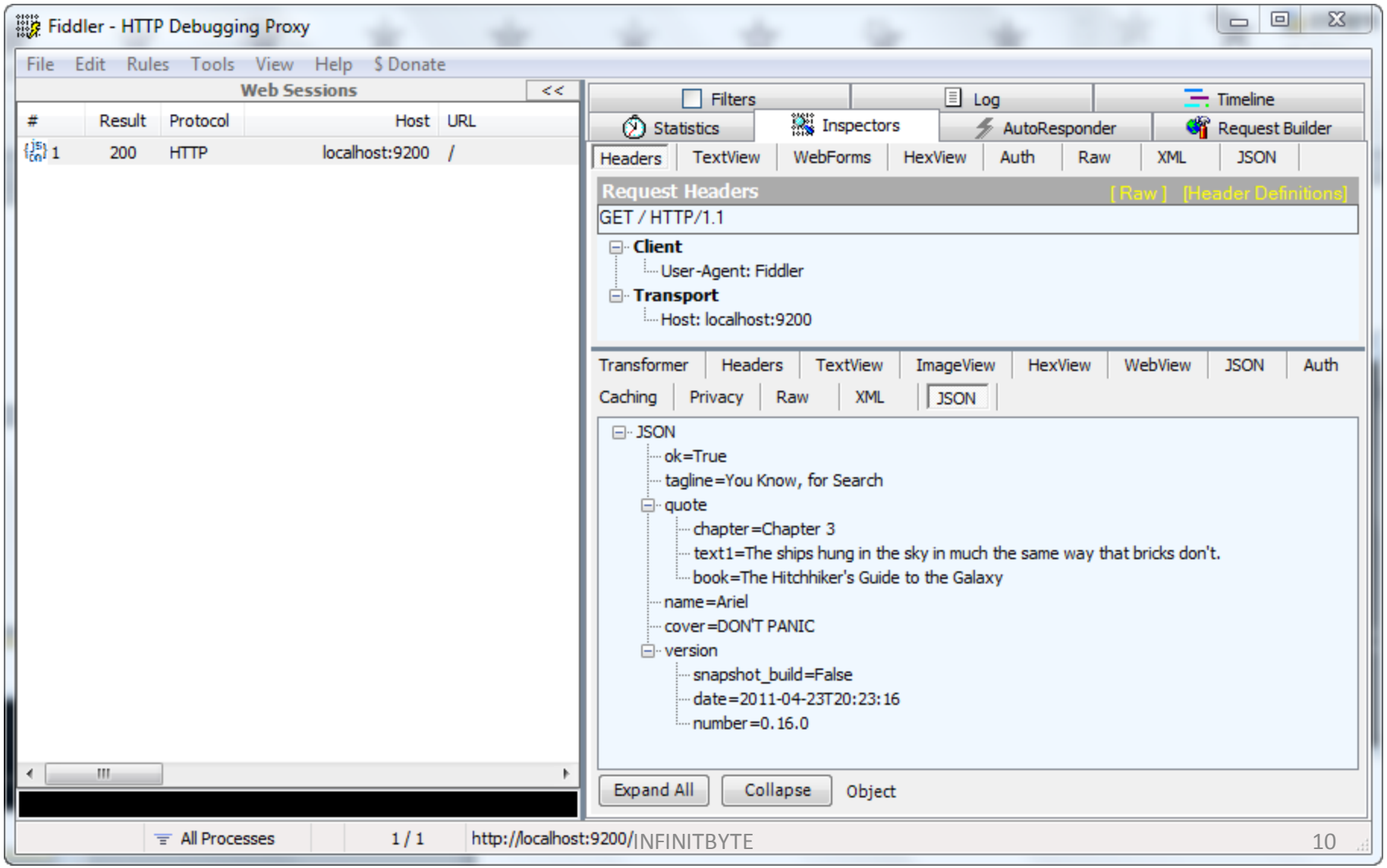
- 先让它跑起来，然后再让它跑快

RTF

- RTF = elasticsearch ready to fly
- ES中文集成包
  - 包含常用插件
    - 服务封装
    - 中文分词
      - analysis-ik
      - analysis-mmseg
      - analysis-pinyin
      - analysis-paoding
      - analysis-smartcn
      - analysis-ansj
    - mapper-attachments
    - transport-thrift
    - tools.carrot2
    - ...
  - 默认做好各种配置

https://github.com/medcl/elasticsearch-rtf
同步更新到最新的0.20.2

Debug Tool: Fiddler



确认ES正常运行

- http，9200端口是否监听
  - netstat –ano|grep 9200
- es通讯端口9300端口是否监听
  - netstat –ano|grep 9300
- 如果使用了thrift，确认9500端口是否监听
  - netstat –ano|grep 9500
- 查看ES进程
  - ps –aux|grep elasticsearch
- 通过elasticsearch-wrapper来查看
  - /etc/init.d/elasticsearch status
- 通过ES的API来查看状态
  - cluster status,node status,index status etc.

查看ES版本信息

curl http://localhost:9200

```
{
  "ok" : true,
  "status" : 200,
  "name" : "Captain Savage",
  "version" : {
    "number" : "0.20.2",
    "snapshot_build" : false
  },
  "tagline" : "You Know, for Search"
}
```

创建索引

- 操作流程
  1. 准备索引文档
     - json格式
       - 自己拼json格式
       - 使用第三方插件，如jdbc river
       - 注意验证json格式有效性
  2. 提交文档到es
     - index
     - bulk
  3. 返回操作结果

     成功或者失败

Index 、 Type、 Doc

- Doc
  - 索引文档，你的数据
- Index
  - 索引库
  - 物理隔离
- Type
  - 索引类型
  - 同类型数据

Explain the url

这个url就是这条索引记录的唯一标示

Index a document

```
curl –XPOST http://localhost:9200/myindex/share/1
-d’
{
  "url" : "http://www.elasticsearch.cn/",
  "date" : "2013-01-20 13:00:00",
  "location" : "beijing,北京"
}’
```

Field
字段名称

字段内容

RESTful
URL地址

索引文档内容，
Json格式

Index Response


```
{
    "ok": true,
    "_index": "myindex",
    "_type": "share",
    "_id": "1",
    "_version": 1
}
```

get the document

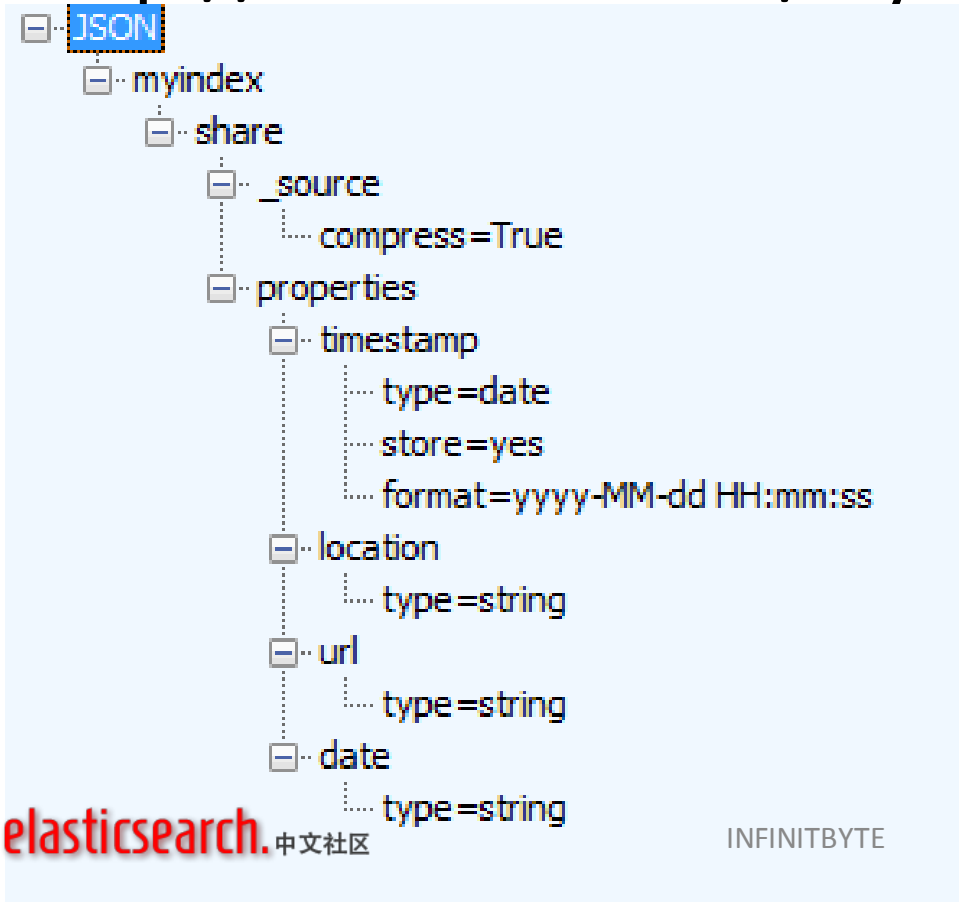
curl –XGET http://localhost:9200/myindex/share/1


```
{
  "_index": "myindex",
  "_type": "share",
  "_id": "1",
  "_version": 1,
  "exists": true,
  "_source": {
    "url": "http://www.elasticsearch.cn/",
    "date": "2013-01-20 13:00:00",
    "location": "beijing,北京"
  }
}
```


Source就是前面索引时提交的json格式的原始文档内容

get mapping

- curl –XGET http://localhost:9200/myindex/_mapping



Bulk Index

- curl –XPOST http://localhost:9200/_bulk

Meta描述

内容

```
{ "index" : { "_index" : "index_1231231231", "_type" : "type", "_id" : "1" } }
{"name":"jack","age":25}
{ "delete" : { "_index" : "index_1231231231", "_type" : "type", "_id" : "2" } }
{ "index" : { "_index" : "index_1231231231", "_type" : "type", "_id" : "3" } }
{"name":"jack","age":25}
```

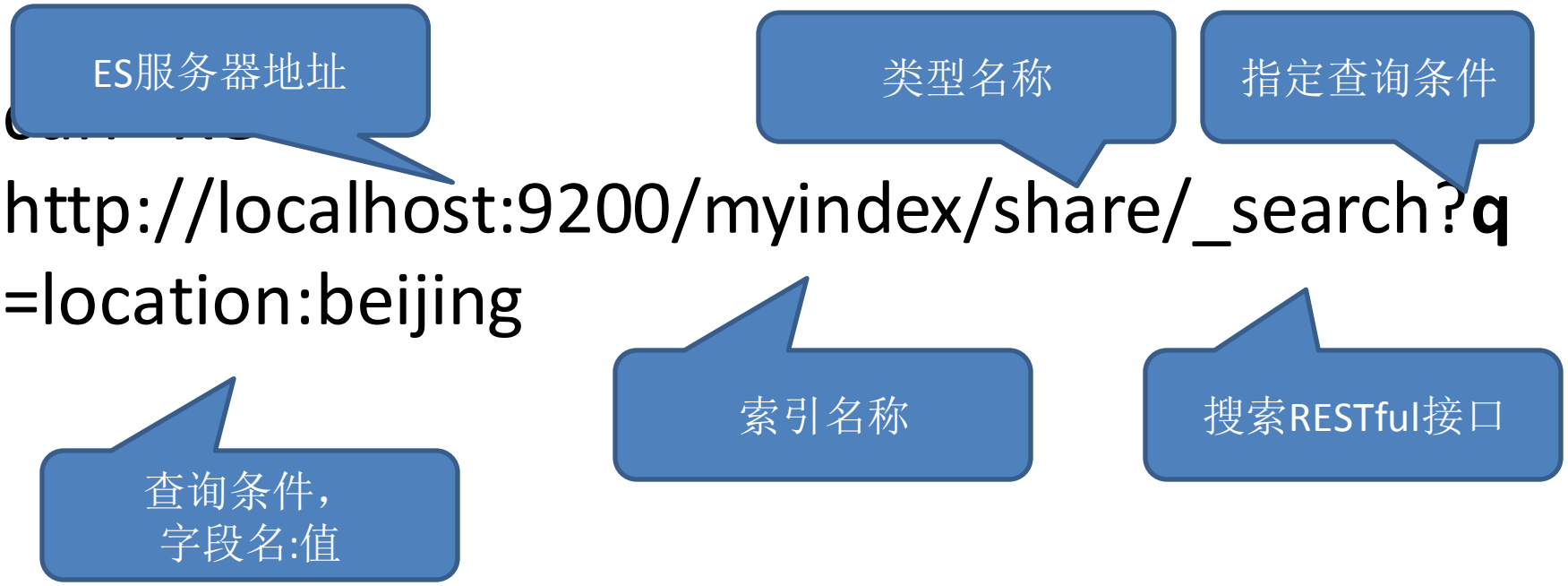
Response of Bulk Index


```
{   "took": 10,   "items": [      {
        "index": {
          "_index": "index_1231231231",
          "_type": "type",
          "_id": "1",
          "_version": 2,
          "ok": true
        }      },      {
        "delete": {
          "_index": "index_1231231231",
          "_type": "type",
          "_id": "2",
          "_version": 1,
          "ok": true
        }      },      {          "index": {
          "_index": "index_1231231231",
          "_type": "type",
          "_id": "3",
          "_version": 1,
          "ok": true
```

构造查询

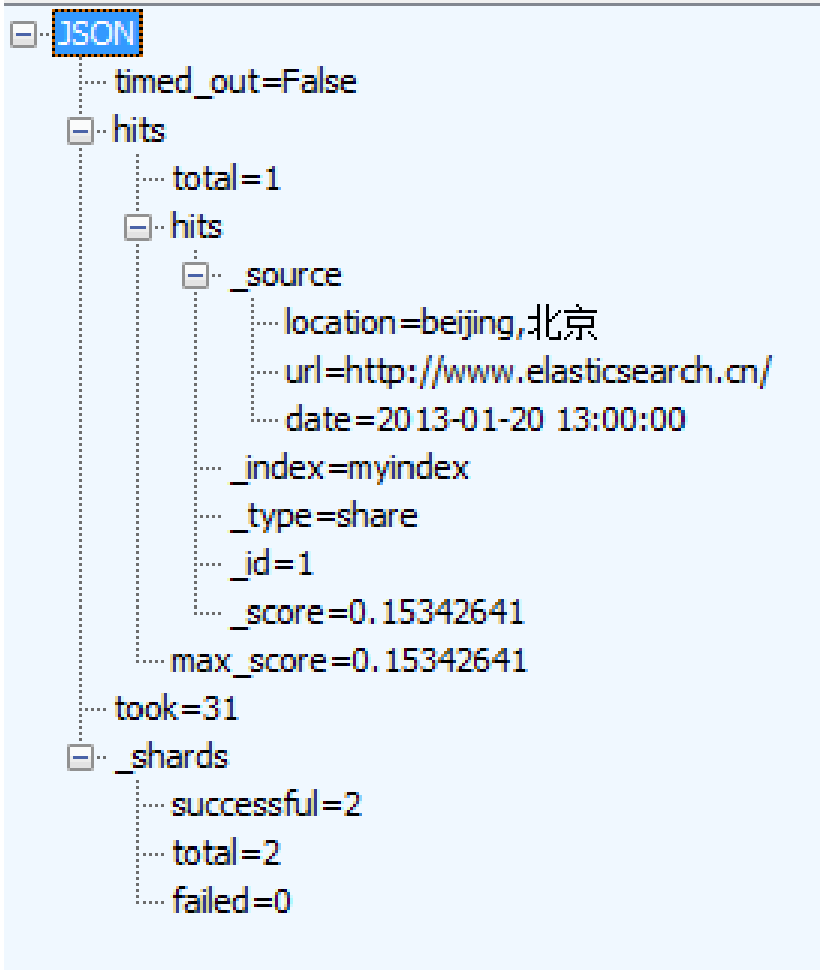
- 查询流程
  1. 构造查询条件
     - lucene查询语法
     - QueryDSL查询表达式
  2. 提交给elasticsearch
  3. 返回搜索结果

Query the document



q后面=的是lucene语法的查询表达式，中文需urlencode，特殊字符需转义，详情：
http://lucene.apache.org/core/3_6_2/queryparsersyntax.html

Search Response



lucene查询语法

- 关键字：+ - && || ! ( ) { } [ ] ^ " ~ * ? : \
  - 如果所要查询的查询词中本身包含关键字，则需要用\进行转义
- 查询词(Term)
  - Lucene支持两种查询词，一种是单一查询词，如"hello"，一种是词组(phrase)，如"hello world"。
- 查询域(Field)
  - 在查询语句中，可以指定从哪个域中寻找查询词，如果不指定，则从默认域中查找。
  - 查询域和查询词之间用:分隔，如title:"Do it right"。

lucene查询语法

- 通配符查询(Wildcard)
  - 支持两种通配符：?表示一个字符，*表示多个字符。
  - 通配符可以出现在查询词的中间或者末尾，如te?t，test*，te*t，但决不能出现在开始，如*test，?test。性能低下！
  - 模糊查询(Fuzzy)
    - 模糊查询的算法是基于Levenshtein Distance，也即当两个词的差别小于某个比例的时候，就算匹配，如roam~0.8，即表示差别小于0.2，相似度大于0.8才算匹配。

lucene查询语法

- 临近查询(Proximity)
  - 在词组后面跟随~10，表示词组中的多个词之间的距离之和不超过10，则满足查询。
  - 所谓词之间的距离，即查询词组中词为满足和目标词组相同的最小移动次数。
  - 如索引中有词组"apple boy cat"。
  - 如果查询词为"apple boy cat"~0，则匹配。
  - 如果查询词为"boy apple cat"~2，距离设为2方能匹配，设为1则不能匹配。
  - 如果查询词为"cat boy apple"~4，距离设为4方能匹配。

lucene查询语法

- 区间查询(Range)
  - 区间查询包含两种，一种是包含边界，用[A TO B]指定，一种是不包含边界，用{A TO B}指定。
  - 如date:[20020101 TO 20030101]，当然区间查询不仅仅用于时间，如title:{Aida TO Carmen}
- 设置查询词的权重(Boost)
  - 可以在查询词后面加^N来设定此查询词的权重，默认是1，如果N大于1，则说明此查询词更重要，如果N小于1，则说明此查询词更不重要。
  - 如jakarta^4 apache，"jakarta apache"^4 "Apache Lucene"

lucene查询语法

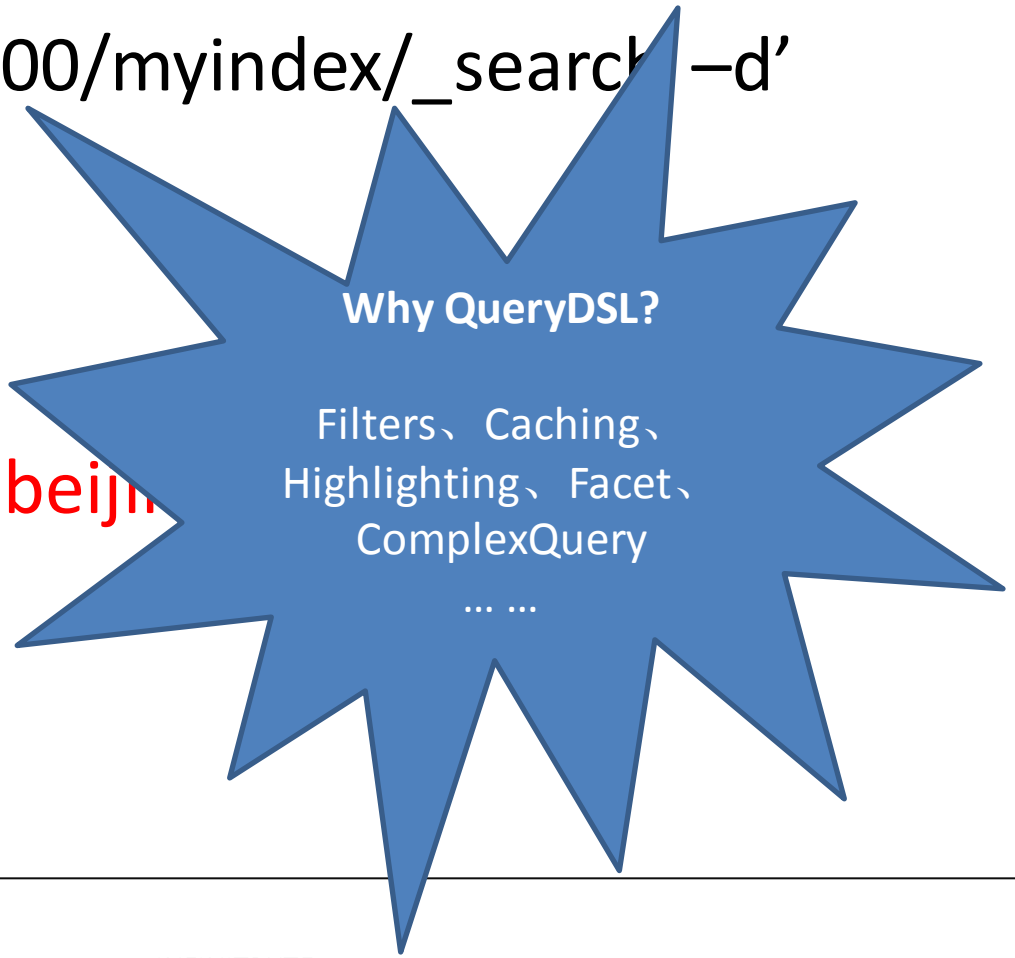
- 布尔操作符（Boolean Operators）
  - 布尔操作符包括连接符，如AND，OR，和修饰符，如NOT，+，-。
  - 默认状态下，空格被认为是OR的关系，QueryParser.setDefaultOperator(Operator.AND)设置为空格为AND。
  - +表示一个查询语句是必须满足的(required)，NOT和-表示一个查询语句是不能满足的(prohibited)。

lucene查询语法

- 灵活组合，支持级联
  - 可以用括号，将查询语句进行组合，从而设定优先级。
    - 如(jakarta OR apache) AND website
  - 字段组合，将多个条件组合到一个字段域下面
    - 如title:(+return +"pink panther")
  - Lucene的查询语法是由QueryParser来进行解析，从而生成查询对象的。
  - 通过编译原理我们知道，解析一个语法表达式，需要经过词法分析和语法分析的过程，也即需要词法分析器和语法分析器。
  - QueryParser是通过JavaCC来生成词法分析器和语法分析器的。

QueryDSL

```
curl  -XPOST
http://localhost:9200/myindex/_search –d’
{
   "query": {
      "term": {
         "location": "beiji
      }
   }
}’
```



Queries

- match
- multi_match
- bool
- boosting
- ids
- custom_score
- custom_boost_factor
- constant_score
- dis_max
- field
- filtered
- flt
- flt_field
- fuzzy
- has_child

match_all
mlt
mlt_field
prefix
query_string
range
span_first
span_near
span_not
span_or
span_term
term
terms
top_children
wildcard
nested
custom_filters_score
indices
text

Filters

- and
- bool
- exists
- ids
- limit
- type
- geo_bbox
- geo_distance
- geo_distance_range
- geo_polygon
- has_child

match_all
missing
not
numeric_range
or
prefix
query
range
script
term
terms
nested

Queries vs Filters

- full text & terms
- relevance scoring
- slower
- no caching

- terms only
- no scoring
- faster
- cacheable

Use filters for anything that doesn't affect the relevance score!

http://slideshare.net/clintongormley/terms-of-endearment-the-elasticsearch-query-dsl-explained

QueryDSL TOPLEVEL

```
{
explain: BOOL,
from: INT,
size: INT,
fields: ["field_1", "field_n"],
query: { QUERY_CLAUSE },
facets: { FACETS_CLAUSE },
sort: { SORT_CLAUSE }
}
```

QUERY_CLAUSE

```
{
term: { TERM_QUERY }
| range: { RANGE_QUERY }
| prefix: { PREFIX_QUERY }
| wildcard: { WILDCARD_QUERY }
| matchAll: { MATCH_ALL_QUERY }
| queryString: { QUERY_STRING_QUERY }
| bool: { BOOLEAN_QUERY }
| disMax: { DISMAX_QUERY }
| constantScore: { CONSTANT_SCORE_QUERY }
| filteredQuery: { FILTERED_QUERY },
}
```

FILTER_CLAUSE

```
{
query: { QUERY_CLAUSE },
| term: { TERM_FILTER },
| range: { RANGE_FILTER },
| prefix: { PREFIX_FILTER },
| wildcard: { WILDCARD_FILTER },
| bool: { BOOLEAN_FILTER },
| constantScore: { CONSTANT_SCORE_QUERY }
}
```

FACETS_CLAUSE

```
{
$facet_name_1:
 { FILTER_CLAUSE },
$facet_name_n: ...
}
```

SORT_CLAUSE

```
{
$fieldname_1
| "score" : {}
| { reverse: BOOL }, $fieldname_n
| "score" : ...
}
```

BOOLEAN_FILTER

```
{
 must:
     { FILTER_CLAUSE }
       | [  { FILTER_CLAUSE }, ... ],
should:
       { FILTER_CLAUSE }
       | [ { FILTER_CLAUSE }, ... ],
 mustNot:
       { FILTER_CLAUSE }
       | [ { FILTER_CLAUSE }, ... ],
 minimumNumberShouldMatch: INT
}
```

BOOLEAN_QUERY

```
{
must: { QUERY_CLAUSE}
| [ { QUERY_CLAUSE  }, ... ],
should: { QUERY_CLAUSE}  | [ { QUERY_CLAUSE  }, ... ],
mustNot: { QUERY_CLAUSE} | [ { QUERY_CLAUSE }, ... ],
boost: FLOAT,
 minimumNumberShouldMatch:  INT
 }
```

？ ？ ？

- 1.什么是mapping？
- 2.如何实现实现查询

Index Setting

- $ curl -XPUT http://localhost:9200/kimchy/ -d \
  '
  index :
  store:
  type: memory
  '

  $ curl -XPUT http://localhost:9200/elasticsearch/ -d \
  ' {
  "index" : {
  "number_of_shards" : 2,
  "number_of_replicas" : 3
  }
  }'

分词设置

- elasticsearch.yml [全局配置]

```
index:
  analysis:
    tokenizer:
    filter:
    analyzer:
      lowercase_keyword:
          type: custom
          filter: [standard,lowercase]
          tokenizer: standard
      lowercase_keyword_ngram_min_size1:
      lowercase_keyword_ngram_min_size2:
      lowercase_keyword_ngram:
      lowercase_keyword_without_standard:
      lowercase_whitespace:
      ik:
          alias: [ik_analyzer]
          type: org.elasticsearch.index.analysis.IkAnalyzerProvider
      mmseg:
      comma_spliter:
```

分词设置

```
curl -XPUT http://localhost:9200/medcl/ -d'
{
   "index":{
      "analysis":{
         "analyzer":{
            "pinyin_analyzer":{
               "tokenizer":"my_pinyin",
               "filter":[
                  "standard"
               ]
            }
         },
         "tokenizer":{
            "my_pinyin":{
               "type": "pinyin",
               "first_letter":"none",
               "padding_char":" "
            }
         }
      }
   }
}'
```

作用于单个索引
不需要重启ＥＳ

IK分词

```
cd bin
plugin -install medcl/elasticsearch-analysis-ik/1.1.0

cd ../config wget http://github.com/downloads/medcl/elasticsearch-analysis-ik/ik.zip --no-check-certificate unzip ik.zip rm ik.zip
```

elasticsearch.conf

```
index:
 analysis:
  analyzer:
    ik:
      alias: [ik_analyzer]
      type: org.elasticsearch.index.analysis.IkAnalyzerProvider
```

其它中文分词，参见：https://github.com/medcl/elasticsearch-rtf

Mapping

- 定义索引下Type的字段处理规则
  - 索引如何建，数据类型
  - 是否保存原始索引json文档
  - 是否压缩原始json文档
  - 是否需要分词处理
  - 如何进行分词处理
    - 索引时怎么分词
    - 查询时怎么分词
  - Routing规则？Parent-Child关系？
  - 。 。 。

创建一个Mapping

```
curl -XPOST http://localhost:9200/medcl/folks/_mapping -d'{
   "folks":{
      "properties":{
         "name":{
            "type": "multi_field",
            "fields":{
               "name":{
                  "type": "string",
                  "store":"no",
                  "term_vector":"with_positions_offsets",
                  "analyzer":"pinyin_analyzer",
                  "boost":10
               },
               "primitive":{
                  "type": "string",
                  "store":"yes",
                  "analyzer":"keyword"
               }        }      }    }}}'
```

内置字段

- _uid
- _id
- _type
- _source
- _all
- _analyzer
- _boost
- _parent
- _routing
- _index
- _size
- _timestamp
- _ttl

系统保留，你就不要用了

字段类型

- string,
- integer/long,
- float/double,
- boolean,
- null.
- date类型（内部存储还是long类型）
  - “format” : “YYYY-MM-dd”

mapping参数

- store：yes、no
- index：analyzed 、not_analyzed 、no
- analyzer
  - index_analyzer
  - search_analyzer
- include_in_all
- omit_term_freq_and_positions
- omit_norms
- null_value
- boost
- term_vector

MappingType

- core
- array
- object
- root object
- nested
- multi_field
- ip
- geo_point
- attachment

Core

```
{
   "tweet" : {
      "properties" : {
         "user" : {"type" : "string", "index" : "not_analyzed"},
         "message" : {"type" : "string", "null_value" : "na"},
         "postDate" : {"type" : "date"},
         "priority" : {"type" : "integer"},
         "rank" : {"type" : "float"}
      }
   }
}
```

Array

```
{
    {
      "tweet" : {
        "properties" : {
          "message" : {"type" : "string"},
          "tags" : {"type" : "string", "index_name" : "tag"},
          "lists" : {
            "properties" : {
              "name" : {"type" : "string"},
              "description" : {"type" : "string"}
            }
          }
        }
      }
    }
}
```

Object

```
{
  "tweet" : {
    "properties" : {
      "person" : {
        "type" : "object",
        "properties" : {
          "name" : {
            "properties" : {
              "first_name" : {"type" : "string"},
              "last_name" : {"type" : "string"}
            }
          },
          "sid" : {"type" : "string", "index" : "not_analyzed"}
        }
      }
      "message" : {"type" : "string"}
    }
  }
}
```

ik分词mapping实例

```
curl -XPOST http://localhost:9200/index/fulltext/_mapping -d'
{
    "fulltext":{
        "_all":{
            "indexAnalyzer":"ik",
            "searchAnalyzer":"ik",
            "term_vector":"no",
            "store":"false"
        },
        "properties":{
            "content":{
                "type": "string",
                "store":"no",
                "term_vector":"with_positions_offsets",
                "indexAnalyzer":"ik",
                "searchAnalyzer":"ik",
                "include_in_all":"true",
                "boost":8
            }
        }
    }
}'
```

Dynamic Mapping

- config\map

```
{
  "_default_": {
    "_source": {
      "compress": true
    },
    "_all":  {
      "enabled": true,
      "store": "no",
      "term_vector": "with_positions_offsets",
      "indexAnalyzer":  "mmseg",
      "searchAnalyzer":  "mmseg"
    },
    "properties":{
      "title": {
        "type": "multi_field",
        "fields": {
          "title": {
            "type": "string",
            "store": "yes",
            "term_vector": "with_positions_offsets",
            "indexAnalyzer":  "mmseg",
            "searchAnalyzer":  "mmseg",
            "include_in_all":  "true",
            "boost": 8
          },
          "pinyin": {
            "type": "string",
            "store": "no",
            "term_vector": "with_positions_offsets",
            "analyzer":  "pinyin_ngram_analyzer",
            "boost": 10
          }
        }
      },
      "desc": {
        "type": "string",
        "store":"yes",
        "term_vector": "with_positions_offsets",
        "indexAnalyzer": "mmseg",
        "searchAnalyzer": "mmseg",
        "include_in_all":  "true",
        "boost":7
      }
    },
    "_routing": {
      "required":true
    },
    "dynamic_templates":  [
      {
        "template_external_field":  {
          "match":  "ext_*",
          "mapping":  {
            "type": "multi_field",
            "fields": {
              "{name}": {
                "type": "{dynamic_type}",
                "store": "yes",
                "term_vector":  "with_positions_offsets",
                "indexAnalyzer":  "mmseg",
                "searchAnalyzer":  "mmseg",
                "include_in_all":  "true",
                "boost": 10
              },
              "ngram": {
                "type": "{dynamic_type}",
                "omit_norms":  false,
                "omit_term_freq_and_positions":  false,
                "term_vector":  "with_positions_offsets",
                "boost": 5,
                "include_in_all":  false,
                "index":  "analyzed",
                "indexAnalyzer":  "lowercase_keyword_ngram_min_size2",
                "searchAnalyzer":  "lowercase_keyword_ngram_min_size2",
                "store": "no"
              }
            }
          }
        }
      }
    ]
  }
}
```

修改、删除Mapping

- 修改，API：POST-》PUT
- 注意：如果类型变化则会造成修改失败，尽量避免修改现有字段，可以动态新增字段定义
- 删除

```
curl -XDELETE http://localhost:9200/medcl/folks/_mapping
```

注意：删除Mapping，其对应type下所有数据也会被删除

Template

```
http://10.22.1.12:9200/_template/app_log_template
{
   “template”: “log*",
   "order": 0,
   "settings": {
      "number_of_shards": 1,
      "number_of_replicas": 1
   }
}
```

Template

```
curl -XPOST http://10.129.8.58:9200/_template/beisen_recruit_template -d'{
    "mappings": {
      "Resume": {
        "_source": {
          "compress": true
        },
        "_all": {
          "enabled": false
        },
        "properties": {
          "Age": {
            "type": "long"
          },
          "UserID": {
            "include_in_all": false,
            "type": "integer"
          }
        }
      },
      "FullText": {
        "_source": {
          "enabled": false
        },
        "_all": {
          "indexAnalyzer": "recruit_fulltext_index_analyzer",
          "searchAnalyzer": "recruit_fulltext_search_analyzer",
          "term_vector": "no",
          "store": "false"
        },
        "properties": {
          "identity": {
            "type": "string",
            "store": "yes",
            "index": "not_analyzed",
            "include_in_all": "false"
          },
          "indextime": {
            "omit_term_freq_and_positions": true,
            "include_in_all": false,
            "omit_norms": true,
            "type": "string",
            "store": "yes",
            "indexAnalyzer": "recruit_fulltext_index_analyzer",
            "searchAnalyzer": "recruit_fulltext_search_analyzer",
            "index": "analyzed"
          }
        },
        "_routing": {
          "required": true
        },
        "_parent": {
          "type": "Resume"
        }
      }
    },
    "template": "index_prefix*",
    "order": 0,
    "settings": {
      "number_of_shards": 4,
      "number_of_replicas": 1
    }
}'
```

实例讲解

实战指导

“社保医院搜索”

G16

	A	B	C	D	E
1	编 码	医院名称	所属区县	医院类别	医院等级
2	14110002	北京市昌平区沙河医院(14110002)	昌平区	对外综合	二级合格
3	14110003	北京市昌平区南口医院(14110003)	昌平区	对外综合	二级合格
4	14110004	北京市昌平区红十字会北郊医院(14110004)	昌平区	对外综合	二级合格
5	14110023	北京市昌平区南口铁路医院(14110023)	昌平区	对外综合	二级合格
6	14155001	北京市昌平区东小口社区卫生服务中心(14155001)	昌平区	对外综合	二级合格
7	14153003	北京市昌平区精神卫生保健院(14153003)	昌平区	对外专科	二级合格
8	14155003	北京皇城股骨头坏死专科医院(14155003)	昌平区	对外专科	二级合格
9	14110001	北京市昌平区医院(14110001)	昌平区	对外综合	二级甲等
10	14152001	北京市昌平区妇幼保健院(14152001)	昌平区	对外专科	二级甲等
11	14153002	北京民康医院（精神病、老年病专科）(14153002)	昌平区	对外专科	二级甲等
12	14151001	北京市昌平区中医医院(14151001)	昌平区	对外中医	二级甲等
13	14110019	北京小汤山医院(14110019)	昌平区	对外综合	三级合格
14	14110027	小汤山医院二部(14110027)	昌平区	对外综合	三级合格
15	14153001	北京回龙观医院（精神病专科）(14153001)	昌平区	对外专科	三级甲等
16	14162001	昌平区区医院昌盛园社区卫生服务站(14162001)	昌平区	社区卫生站	未评级
17	14162002	昌平区城区永安社区卫生服务站(14162002)	昌平区	社区卫生站	未评级
18	14162003	昌平区城区东关南里社区卫生服务站(14162003)	昌平区	社区卫生站	未评级
19	14162005	北京市昌平区中医医院园区社区卫生服务站(14162005)	昌平区	社区卫生站	未评级
20	14162006	西关社区卫生服务站(14162006)	昌平区	社区卫生站	未评级
21	14162007	北门社区卫生服务站(14162007)	昌平区	社区卫生站	未评级
22	14162008	郝庄社区卫生服务站(14162008)	昌平区	社区卫生站	未评级
23	14162009	亢山广场社区卫生服务站(14162009)	昌平区	社区卫生站	未评级
24	14162010	北环社区卫生服务站(14162010)	昌平区	社区卫生站	未评级
	14162011	北京市昌平区回龙观社区卫生服务中心龙华园社区卫生服	昌平区	社区卫生站	未评级

需求

- 1.通过编码快速查找 能
- 2.通过医院名称模糊搜索 能
- 3.可按医院类别、区县、等级过滤 能
- 4.能通过医院的pinyin搜索医院 能
- 5.能分组显示类别、区县、等级 能
- 6.能买到回家的火车票吗 不能

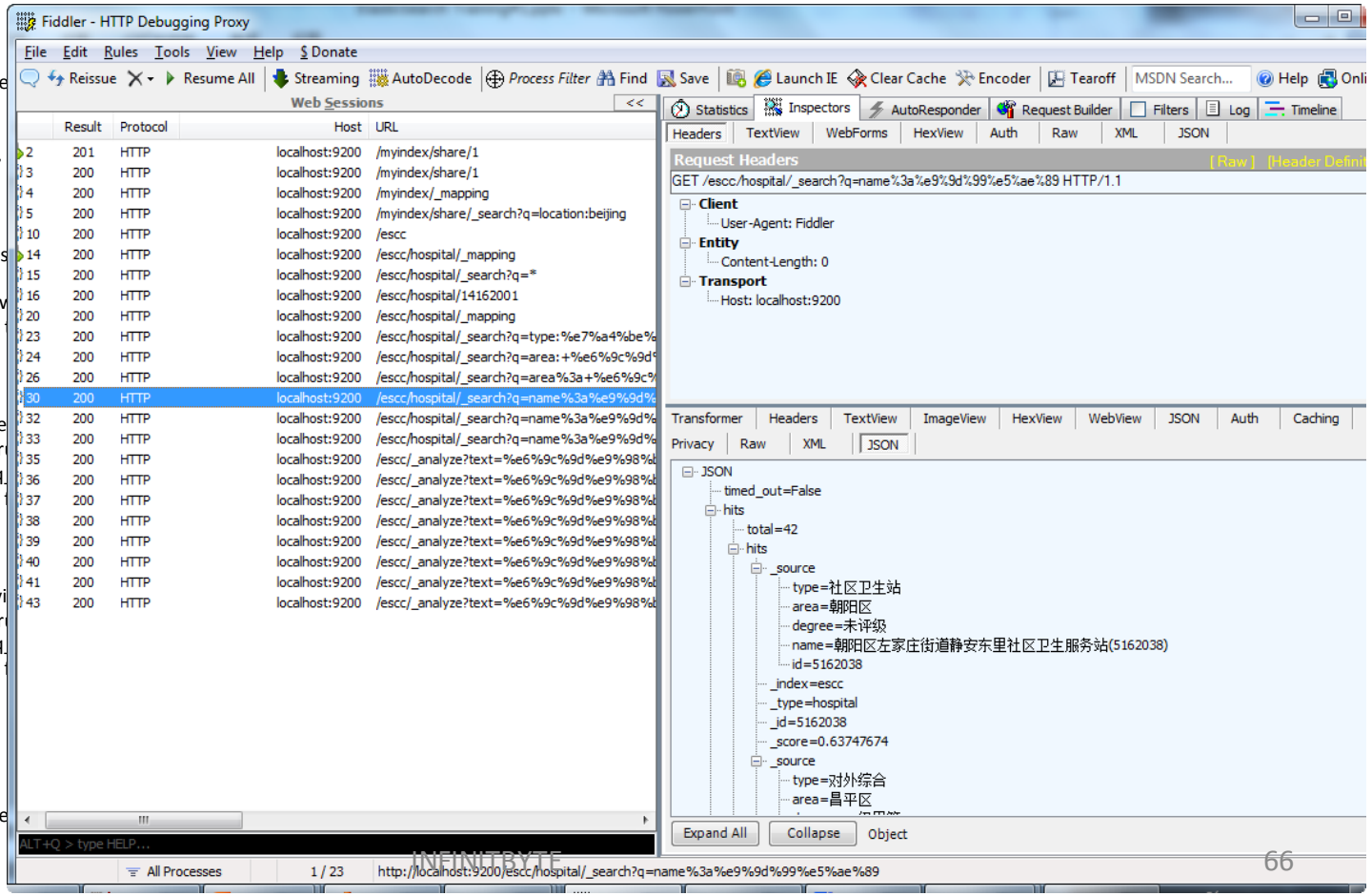
Mapping

字段	类型	描述
Id	Integer	编码，不需要分词
Name	String	医院名称，需要中文分词和拼音分词
Area	String	区县，不需要分词
Type	String	医院类别，不需要分词
Degree	String	医院等级，不需要分词

Fiddler实战调试演练！

Mapping

```
{
  "hospital": {
    "_source": {
      "compress": "true"
    },
    "properties": {
      "id": {
        "type": "integer",
        "index": "not_analyze
      },
      "name": {
        "type": "multi_field",
        "fields": {
          "name": {
            "type": "string",
            "analyzer": "mms
            "store": "yes",
            "term_vector": "v
            "include_in_all":
          },
          "ngram": {
            "type": "string",
            "analyzer": "lowe
            "omit_norms": tr
            "omit_term_freq
            "include_in_all":
          },
          "pinyin": {
            "type": "string",
            "analyzer": "pinyi
            "omit_norms": tr
            "omit_term_freq
            "include_in_all":
          }
        }
      },
      "area": {
        "type": "string",
        "index": "not_analyze
      },
      "type": {
        "type": "string",
        "index": "not_analyzed"
```



客户端使用

支持的客户端

- ElasticSearch.pm: Perl client.
- pyes: Python client.
- Tire: Ruby API & DSL, with ActiveRecord/ActiveModel integration.
- Elastica: PHP client.
- rubberband: Ruby client.
- em-elasticsearch: elasticsearch library for eventmachine.
- elastic_searchable: Ruby client + Rails integration.
- erlastic_search: Erlang client.
- elasticsearch PHP client.
- PlainElastic.Net: .NET client.
- NEST: .NET client.
- ElasticSearch.NET: .NET client.
- pyelasticsearch: Python client.
- Elastisch: Clojure client.
- ESClient: A lightweight and easy to use Python client for ElasticSearch
- scalastic: Scala client
- rawes: Python low level client
- … …

JAVA-API

Maven repositories

```
<dependency>
        <groupId>org.elasticsearch</groupId>
        <artifactId>elasticsearch</artifactId>
        <version>${es.version}</version>
      </dependency>
```

Client-类型

- Node Client
  - 客户端节点本身也是elasticsearch节点
  - 也加入集群，和其它elasticsearch节点一样
  - 升级维护麻烦（词库，配置等等）
- TransportClient
  - 更加轻量级
  - 客户端socket连接到es集群
  - Netty线程池

TransportClient

```
Map<String, String> m = new HashMap<String, String>();

m.put("cluster.name", “ESCLUSTER");

Settings s = ImmutableSettings.settingsBuilder().put(m).build();

Client client=(Client) new TransportClient(s);

client.addTransportAddress(new InetSocketTransportAddress(HOST1, 9300));

client.addTransportAddress(new InetSocketTransportAddress(HOST2, 9300))
```

```
client.close();
```

Index

```
Map<String,Object> fields=new HashMap<String,
Object>();
fields.put("department",dto.getDepartment());
fields.put("type",dto.getDoctorType().getValue());
fields.put("gender",dto.getGender().getValue());

client.prepareIndex(index, type)
.setId(id)
.setSource(fields)
.execute()
.actionGet();
```

Index

```
import static org.elasticsearch.common.xcontent.XContentFactory.*;

IndexResponse response = client.prepareIndex("twitter", "tweet", "1")
        .setSource(jsonBuilder()
                    .startObject()
                        .field("user", "kimchy")
                        .field("postDate", new Date())
                        .field("message", "trying out Elastic Search")
                    .endObject()
                  )
        .execute()
        .actionGet();
```

Bulk

```
BulkRequestBuilder bulkRequest = client.prepareBulk();
// either use client#prepare, or use Requests# to directly build index/delete requests
bulkRequest.add(client.prepareIndex("twitter", "tweet", "1")
        .setSource(jsonBuilder()
                    .startObject()
                        .field("user", "kimchy")
                        .field("postDate", new Date())
                        .field("message", "trying out Elastic Search")
                    .endObject()
                  )
        );
bulkRequest.add(client.prepareIndex("twitter", "tweet", "2")
        .setSource(jsonBuilder()
                    .startObject()
                        .field("user", "kimchy")
                        .field("postDate", new Date())
                        .field("message", "another post")
                    .endObject()
                  )
        );

BulkResponse bulkResponse = bulkRequest.execute().actionGet();
```

Search

```
client.prepareSearch(INDEX)
.setSearchType(SearchType.DFS_QUERY_THEN_FETCH)
.setQuery(termQuery("gender", "male"))
.addSort("age", SortOrder.ASC)
.setFrom(0)
.setSize(10)
.setExplain(true)
.execute()
.actionGet();
```

Search

```
import static org.elasticsearch.index.query.FilterBuilders.*;
import static org.elasticsearch.index.query.QueryBuilders.*;

QueryBuilder qb1 = termQuery("name", "kimchy");

QueryBuilder qb2 = boolQuery()
                .must(termQuery("content", "test1"))
                .must(termQuery("content", "test4"))
                .mustNot(termQuery("content", "test2"))
                .should(termQuery("content", "test3"));

QueryBuilder qb3 = filteredQuery(
        termQuery("name.first", "shay"),
        rangeFilter("age")
            .from(23)
            .to(54)
            .includeLower(true)
            .includeUpper(false)
        );
```

Search

```
BoolQueryBuilder qb2= QueryBuilders.boolQuery();
    BoolQueryBuilder innerBq=boolQuery();

    if(specialty!=null&&!specialty.equals("")){
       innerBq.should(textQuery("specialty",specialty)).boost(10);
    }

    if(keyword!=null&&!keyword.equals("")){
       QueryStringQueryBuilder query = queryString(keyword);
       query.field("full_name",50);
       query.field("full_name.ngram",40);
       query.field("full_name.pinyin_first_letter", 10);
       query.field("full_name.pinyin", 2);
       qb2.must(query);
    }
    if(innerBq.hasClauses()){
       qb2.must(innerBq);
    }
```

Search

```
 SearchResponse searchResult =
ElasticSearchHelper.getResult("mcare", "hospital", qb2, 0, 10);
        if (searchResult != null) {
            for (SearchHit hit : searchResult.getHits()) {
                Map<String, Object> fields = hit.getSource();
                result.put(fields.get("full_name").toString(),
hit.getId());
            }
        }
```

Delete

```
DeleteResponse response =
client.prepareDelete("twitter", "tweet", "1")
        .setOperationThreaded(false)
        .execute()
        .actionGet();
```

ElasticSearch.NET

https://github.com/medcl/ElasticSearch.Net

Introduction

- An ElasticSearch .NET Client
- Connection Pool
- Both Http and Thrift Protocol
- Server Failure Detect， Failover&Failback
- Complex Lucene Expression Constructor
- QueryDSL
- Index&BulkIndex
- ……

In Production 2years+

ElasticSearch.config

```
<?xml version="1.0" encoding="utf-8"?>
<ElasticSearchConfig majorVersion="1" minorVersion="18">
  <TransportType>Thrift</TransportType>
  <ConnectionPool PoolSize="50" LifetimeMinutes="60">...</ConnectionPool>
  <ThriftNodes>
    <Node Host="localhost" Port="9500" Enabled="true" IsFramed="false" EnablePool="true"
      <ConnectionPool PoolSize="60" LifetimeMinutes="60">...</ConnectionPool>
    </Node>
    <Node Host="10.1.1.1" Port="9500" Enabled="false" IsFramed="false" EnablePool="true"
    <Node Host="10.1.1.2" Port="9500" Enabled="false" IsFramed="false" EnablePool="false
  </ThriftNodes>
  <HttpNodes>
    <Node Host="localhost" Port="9200" Enabled="true" />
    <Node Host="localdev" Port="9200" Enabled="false" />
  </HttpNodes>
</ElasticSearchConfig>
```

创建客户端

```
var client = new ElasticSearchClient("10.129.1.1",9500,TransportType.Thrift);
```

```
var client = new ElasticSearchClient("localhost");
```

创建索引

```
var index = "index_test_parent_child_type";

var parentType = new TypeSetting("blog");
parentType.AddStringField("title");

var client = new ElasticSearchClient("localhost");

client.CreateIndex(index, new IndexSetting(10, 1));

var op= client.PutMapping(index, parentType);
```

索引

```
var op= client.Index(indexAabbTestOpen,
"type", "key", "{a:1}");
```

```
var result= client.Index("index", "type", "1",
new Dictionary<string, object>() {{"medcl",
"value"}}, "1");
```

Bulk

```
var type = "bulk";
IList<IndexItem> indexItems = new List<IndexItem>();
var indexItem = new IndexItem(type, "kk1");
indexItem.Add("Name", "medcl1");
indexItem.Add("Name", "medcl2");
indexItems.Add(indexItem);

indexItem = new IndexItem(type, "kk2");
indexItem.Add("Name", "网易”);
indexItem.Add("Name", "163");
indexItems.Add(indexItem);

var result = client.Index(app,  indexItems);
```

Query

```
BooleanQuery booleanQuery=new BooleanQuery();

booleanQuery.Add(new TermQuery(new
Term("name","medcl")),BooleanClause.Occur.MUST);
booleanQuery.Add(new TermQuery(new
Term("age","25")),BooleanClause.Occur.MUST);

new ElasticSearch.Client.ElasticSearchClient("localhost")
.Search("index", "type", booleanQuery.ToString());
```

Query

```
var query = new TermQuery("type", "common");
var dict = new Dictionary<string, object>();
dict["param1"] = 0.2;
var script = "_score + doc['age'].value - param1";
var q = new CustomScoreQuery(query, script, dict);
var result = client.Search(index, "type", q, 0, 5);
```

```
var termQuery = new TermQuery("gender", "true");
var queryFilter = new QueryFilter(termQuery);
var constanFilter = new ConstantScoreQuery(queryFilter);
var result2 = client.Search(index, "type" , constanFilter, 0, 5);
```

Conditional

```
count = client.Count(app,  indexType, Conditional.Get(ExpressionEx.Eq("age", 22)));
```

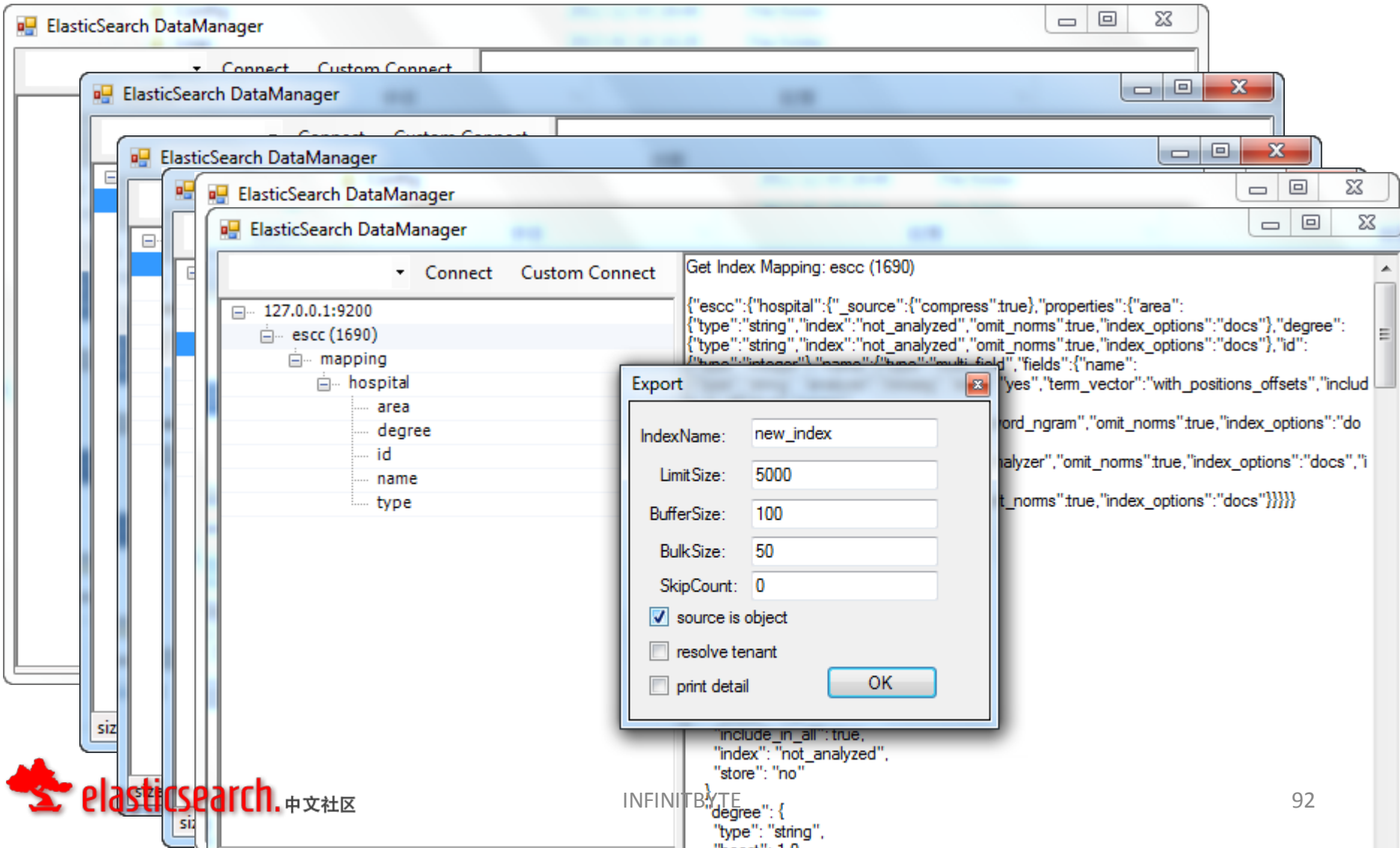
```
var cond1= Conditional.Get(ExpressionEx.Eq("name", "jack"))
    .And(ExpressionEx.Between("age",22,30))
    .And(ExpressionEx.Fuzzy("address","beijing",1.0f,4))
    .And(ExpressionEx.Le("no",87));
Conditional cond2 = Conditional.Or(cond1,
Conditional.Not(ExpressionEx.Eq("gender", "male")));
client.Search("index", "type", cond2.Query);
```

Complex Lucene Query Constructor

Delete

```
client.DeleteIndex(app);
client.DeleteByQueryString("hello", "*");
client.Delete("hello","type", "1");
```

ElasticSearch.DataManager



Q/A

Thank you !


elasticsearch.中文社区


ESCC
ElasticSearch China Conference #1


INFINITBYTE