

**Florian Hopf**  
**[www.florian-hopf.de](http://www.florian-hopf.de)**  
**@fhopf**





elasticsearch.



# Agenda

- Suche
- Verteilung
- Elasticsearch und Java
- Aggregationen
- Zentralisiertes Logging

# Suche

 [Pull requests](#) [Issues](#) [Gist](#)   

Search

---

**Repositories** 5,217  
[Code](#) 474,951  
[Issues](#) 33,100  
[Users](#) 12


**Languages**

Java	918
JavaScript	666
Ruby	591
Python	495
Shell	491
PHP	270
C#	115
Scala	111
Go	102
Puppet	50

[Advanced search](#) [Cheat sheet](#)


**We've found 5,217 repository results** Sort: **Best match** ▾

---


**elastic/elasticsearch** Java ★ 11,589  3,668

Open Source, Distributed, RESTful Search Engine

Updated 5 hours ago




---


**dockerfile/elasticsearch** ★ 212  208

**ElasticSearch** Dockerfile for trusted automated Docker builds.

Updated on 5 May




---


**nervetattoo/elasticsearch** PHP ★ 237  67

Simple PHP client for **ElasticSearch**

Updated on 27 Jan



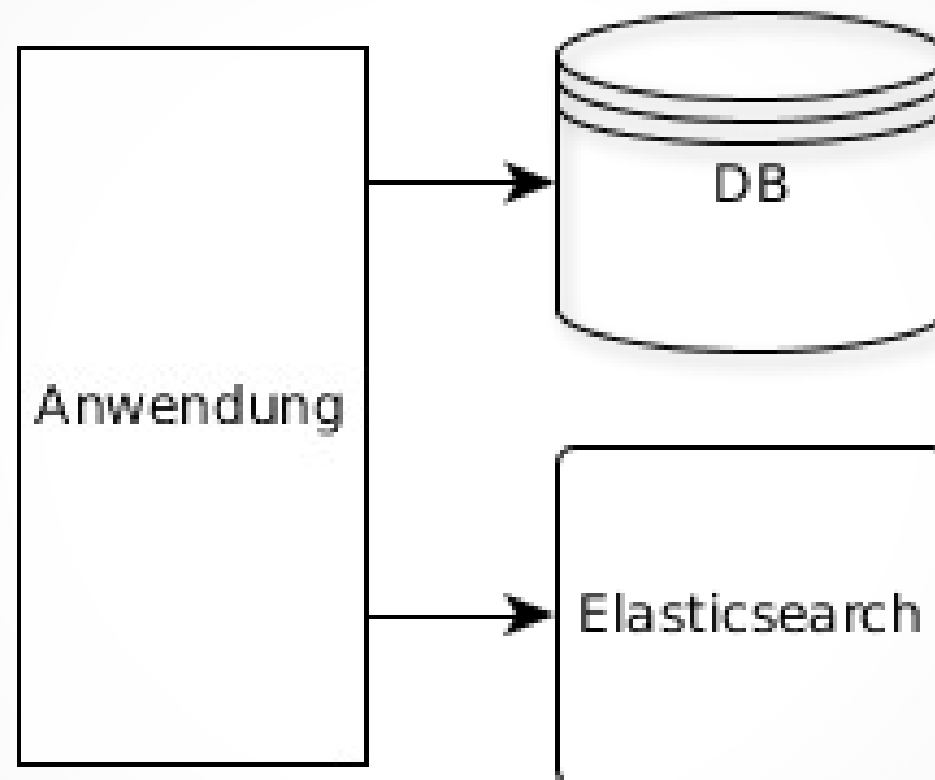
---

**docker-library/elasticsearch** Shell ★ 28  22

Docker Official Image packaging for **elasticsearch**

Updated 3 days ago

# Suche





# Installation

```
# download archive
wget https://download.elastic.co/elasticsearch
      /elasticsearch/elasticsearch-1.7.2.zip

# zip is for windows and linux
unzip elasticsearch-1.7.2.zip

# on windows: elasticsearch.bat
elasticsearch-1.7.2/bin/elasticsearch
```

# Zugriff per HTTP

```
curl -XGET "http://localhost:9200"
```

```
{  
  "status": 200,  
  "name": "Ultron",  
  "cluster_name": "elasticsearch",  
  "version": {  
    "number" : "1.7.2",  
    "build_hash" : "e43676b1385b8125...",  
    "build_timestamp" : "2015-09-14T09:49:53Z",  
    "build_snapshot" : false,  
    "lucene_version" : "4.10.4"  
  },  
  "tagline": "You Know, for Search"  
}
```

# Indizierung

```
POST /library/book
{
  "title": "Elasticsearch in Action",
  "author": [ "Radu Gheorghe",
              "Matthew Lee Hinman",
              "Roy Russo" ],
  "published": "2015-06-30T00:00:00.000Z",
  "publisher": {
    "name": "Manning",
    "country": "USA"
  }
}
```

# Suche

```
POST /library/book/_search?q=elasticsearch"
```

```
{  
  [...]  
  "hits": {  
    "hits": [  
      {  
        "_index": "library",  
        "_type": "book",  
        "_source": {  
          "title": "Elasticsearch in Action",  
          [...]  
        }  
      }  
    ]  
  }  
}
```

# Suche per Query DSL

```
POST /library/book/_search
{
  "query": {
    "filtered": {
      "query": {
        "match": {
          "title": "elasticsearch"
        }
      },
      "filter": {
        "term": {
          "publisher.name": "manning"
        }
      }
    }
  }
}
```

# Sprachspezifische Inhalte

```
POST /library/book/
{
  "title": "Elasticsearch - Ein praktischer Einstieg",
  "author": "Florian Hopf",
  "published": "2015-10-26T00:00:00.000Z",
  "publisher": {
    "name": "dpunkt.verlag",
    "country": "DE"
  },
  "tags": ["Lucene", "Elasticsearch"]
}
```

# Sprachspezifische Inhalte

```
curl -XPOST "http://localhost:9200/library/book/_search"
-d'
{
  "query": {
    "filtered": {
      "query": {
        "match": {
          "title": "praktisch"
        }
      }
    }
  }
}'
```

# Sprachspezifische Inhalte

```
POST /library/book/_search
{
  "query": {
    "filtered": {
      "query": {
        "match": {
          "title": "praktisch"
        }
      }
    }
  }
}
```



# Analyzing

Elasticsearch  
in Action

1. Tokenization

Term	Document Id
Action	1
ein	2
Einstieg	2
Elasticsearch	1,2
in	1
praktischer	2

Elasticsearch:  
Ein praktischer  
Einstieg

# Analyzing

Elasticsearch  
in Action

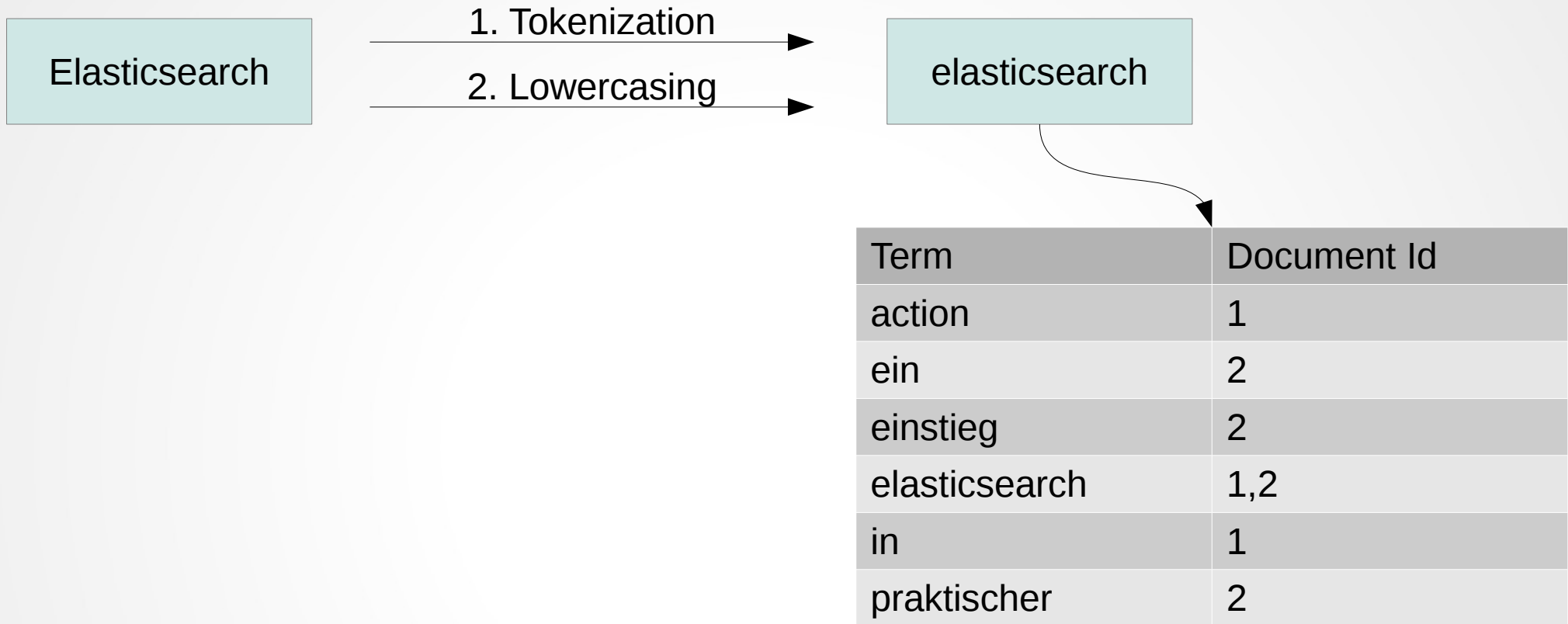
1. Tokenization

2. Lowercasing

Term	Document Id
action	1
ein	2
einstieg	2
elasticsearch	1,2
in	1
praktischer	2

Elasticsearch:  
Ein praktischer  
Einstieg

# Suche



# Suche

praktisch

1. Tokenization



2. Lowercasing



praktisch

Term	Document Id
action	1
ein	2
einstieg	2
elasticsearch	1,2
in	1
praktischer	2

# Analyzing

Elasticsearch  
in Action

1. Tokenization

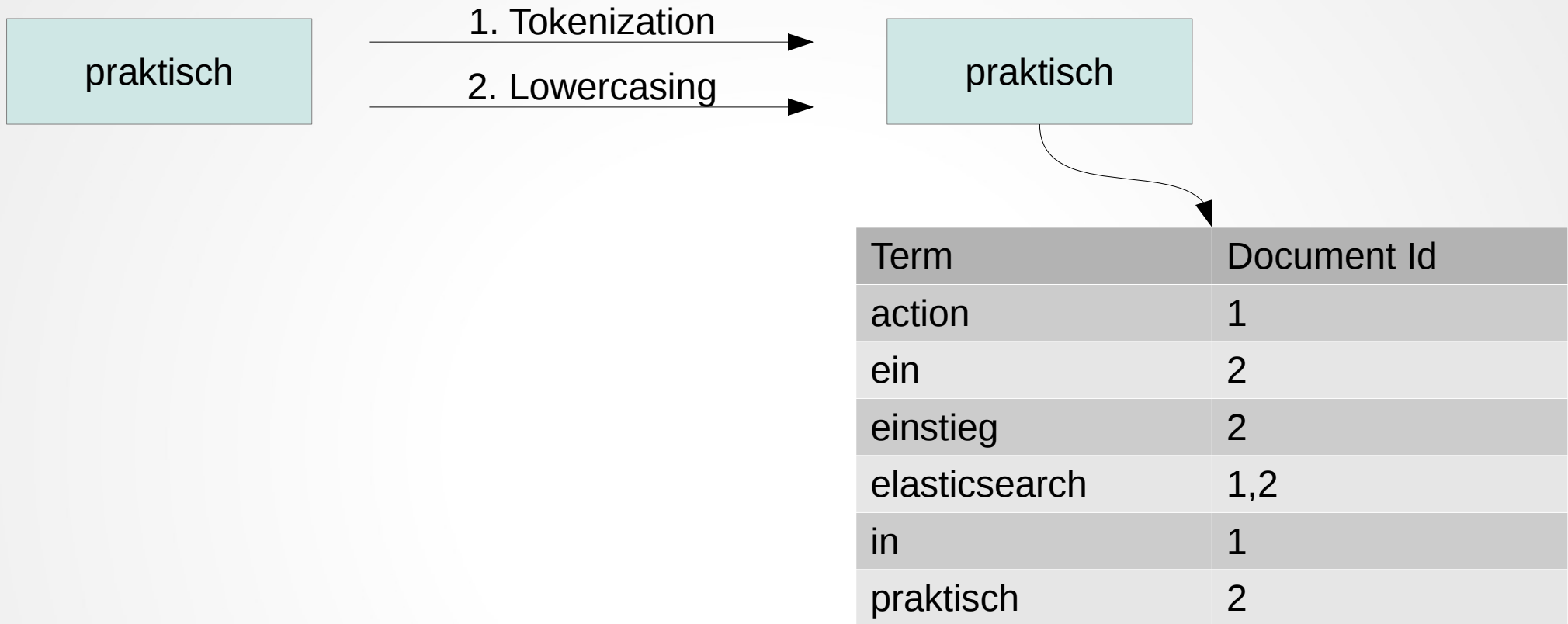
2. Lowercasing

3. Stemming

Term	Document Id
action	1
ein	2
einstieg	2
elasticsearch	1,2
in	1
praktisch	2

Elasticsearch:  
Ein praktischer  
Einstieg

# Suche



# Mapping

```
PUT /library/book/_mapping
{
  "book": {
    "properties": {
      "title": {
        "type": "string",
        "analyzer": "german"
      }
    }
  }
}
```

# Suchfeatures

- Fertige Analyser, Konfiguration von eigenen
- Relevanzberechnung
- Paginierung, Sortierung
- Highlighting, Autovervollständigung, ...
- Facettierung über Aggregationen



# Recap

- Java-basierter Suchserver
- Kommunikation über HTTP und JSON
- Dokumentenbasierte Speicherung
- Unterstützung unterschiedlicher Datentypen
- Suche über Query DSL auf invertiertem Index

# Verteilung

# Verteilung

The screenshot displays a Kibana dashboard for an index named 'library'. The index has 5 shards and 1 document, with a total size of 4.25KB. The dashboard shows a warning for 5 unassigned shards and a table of shard status for node 'Ultron'.

**Index: library**  
shards: 5 \* 2 | docs: 1 | size: 4.25KB

**Warning:** 5 unassigned shards  
*show only unhealthy indices*

**Node: Ultron**  
172.28.4.45:9300

**Shard Status:**

Shard ID	Status
2	Unassigned (Grey)
0	Unassigned (Grey)
3	Unassigned (Grey)
1	Unassigned (Grey)
4	Unassigned (Grey)

**Performance Metrics:**

Metric	Value
heap	Low
disk	High
cpu	Low
load	Low

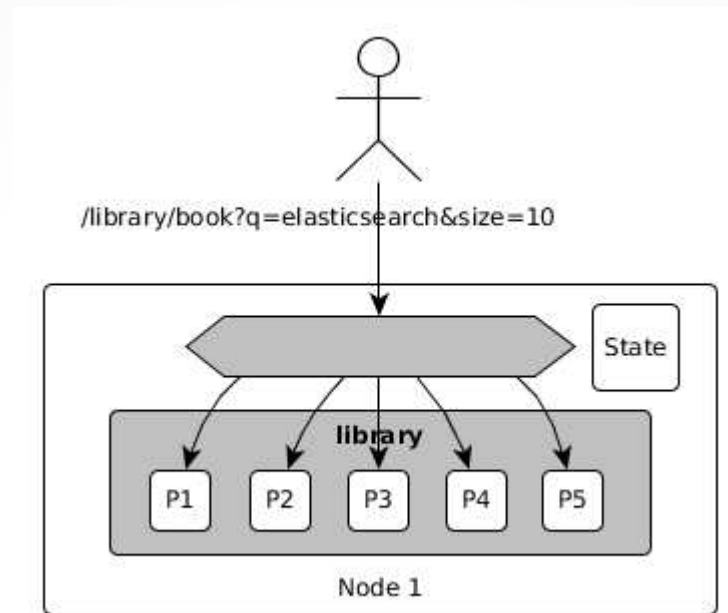
# Verteilung



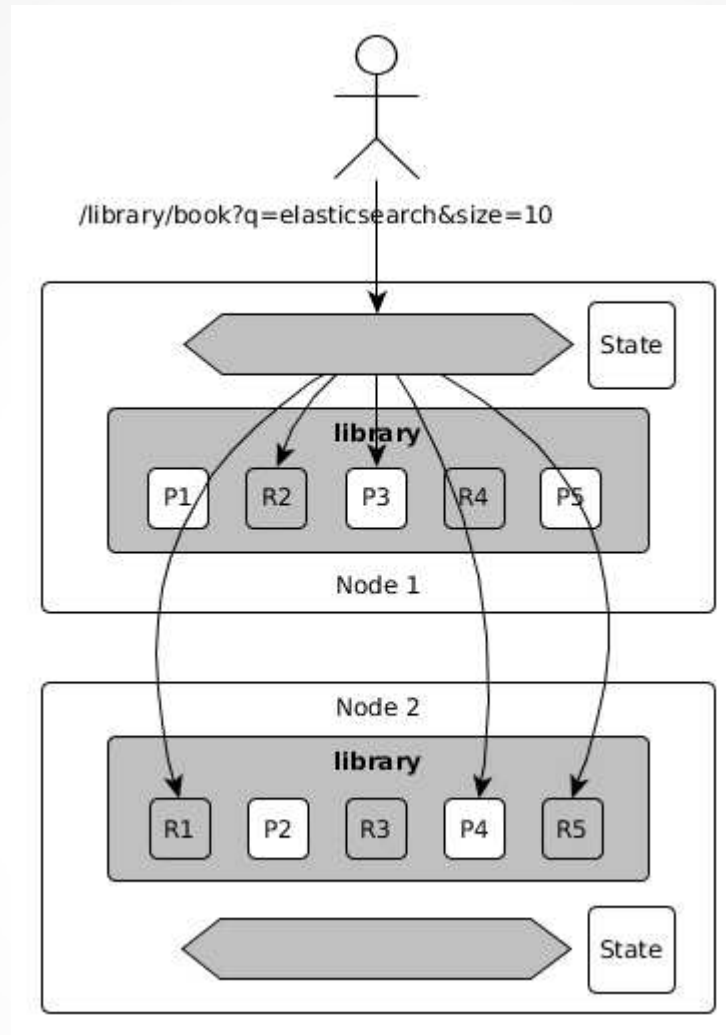
# Verteilung



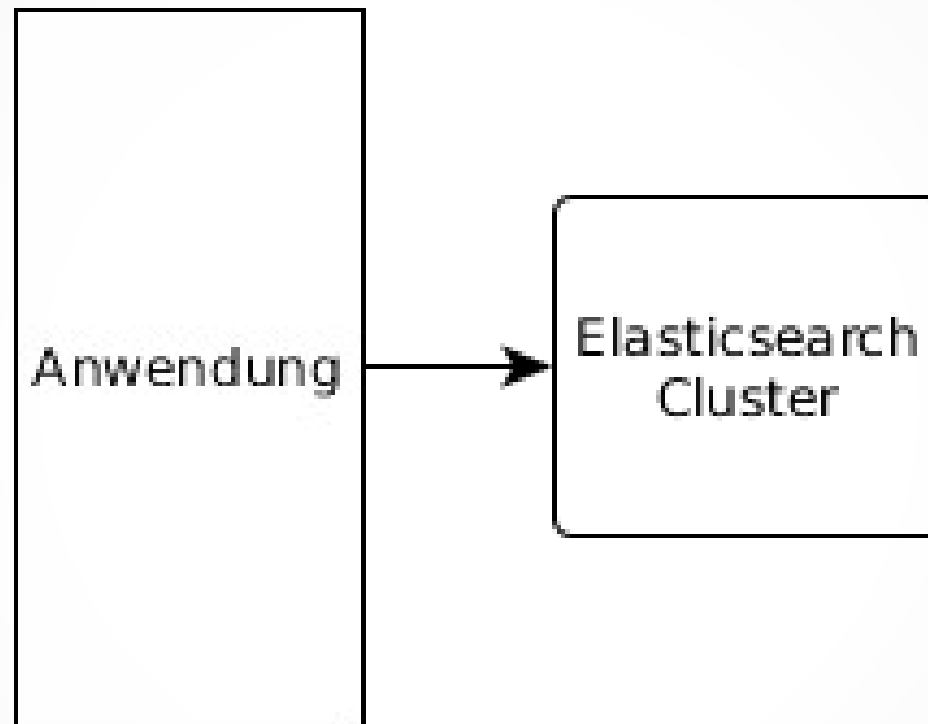
# Suche im Cluster



# Suche im Cluster



# Verteilung





# Recap

- Knoten können Cluster bilden
- Datenverteilung durch Sharding
- Replicas zur Lastverteilung und Ausfallsicherheit
- Verteilte Suche
- Cluster-Zustand auf jedem Knoten

Java

# Java!

```
dependencies {  
    compile group: 'org.elasticsearch',  
            name: 'elasticsearch',  
            version: '1.7.2'  
}
```

# Java!

```
TransportAddress address =  
    new InetSocketAddress("localhost", 9300);  
  
Client client = new TransportClient().  
    addTransportAddress(address);
```

# Suche per Query DSL

```
POST /library/book/_search
{
  "query": {
    "filtered": {
      "query": {
        "match": {
          "title": "elasticsearch"
        }
      },
      "filter": {
        "term": {
          "publisher.name": "manning"
        }
      }
    }
  }
}
```

# Suche über Java-Client

```
SearchResponse searchResponse = client
    .prepareSearch("library")
    .setQuery(filteredQuery(
        matchQuery("title", "elasticsearch"),
        termFilter("publisher.name", "manning")))
    .execute().actionGet();

assertEquals(1, searchResponse.getHits().getTotalHits());













SearchHit hit = searchResponse.getHits().getAt(0);
assertEquals("Elasticsearch in Action",
    hit.getSource().get("title"));
```

# Indizierung über Java-Client

```
XContentBuilder jsonBuilder = jsonBuilder()  
    .startObject()  
        .field("title", "Elasticsearch")  
        .field("author", "Florian Hopf")  
        .startObject("publisher")  
            .field("name", "dpunkt")  
        .endObject()  
    .endObject();  
  
client.prepareIndex("library", "book")  
    .setSource(jsonBuilder)  
    .execute().actionGet();
```

# Indizierung über Java-Client

```
client.prepareIndex("testindex", "book").
```

- m  **setSource** (byte[] source) IndexRequestBuilder
- m  **setSource** (byte[] source, int offset, int ... IndexRequestBuilder
- m  **setSource** (BytesReference source) IndexRequestBuilder
- m  **setSource** (Map<String, Object> source, XCo... IndexRequestBuilder
- m  **setSource** (Object... source) IndexRequestBuilder
- m  **setSource** (String field1, Object value1) IndexRequestBuilder
- m  **setSource** (String field1, Object value1, S... IndexRequestBuilder
- m  **setSource** (String field1, Object value1, S... IndexRequestBuilder
- m  **setSource** (String field1, Object value1, S... IndexRequestBuilder
- m  **setSource** (String source) IndexRequestBuilder
- m  **setSource** (XContentBuilder sourceBuilder) IndexRequestBuilder
- m  **setTimestamp** (String timestamp) IndexRequestBuilder

Dot, semicolon and some other keys will also close this lookup and be inserted into editor



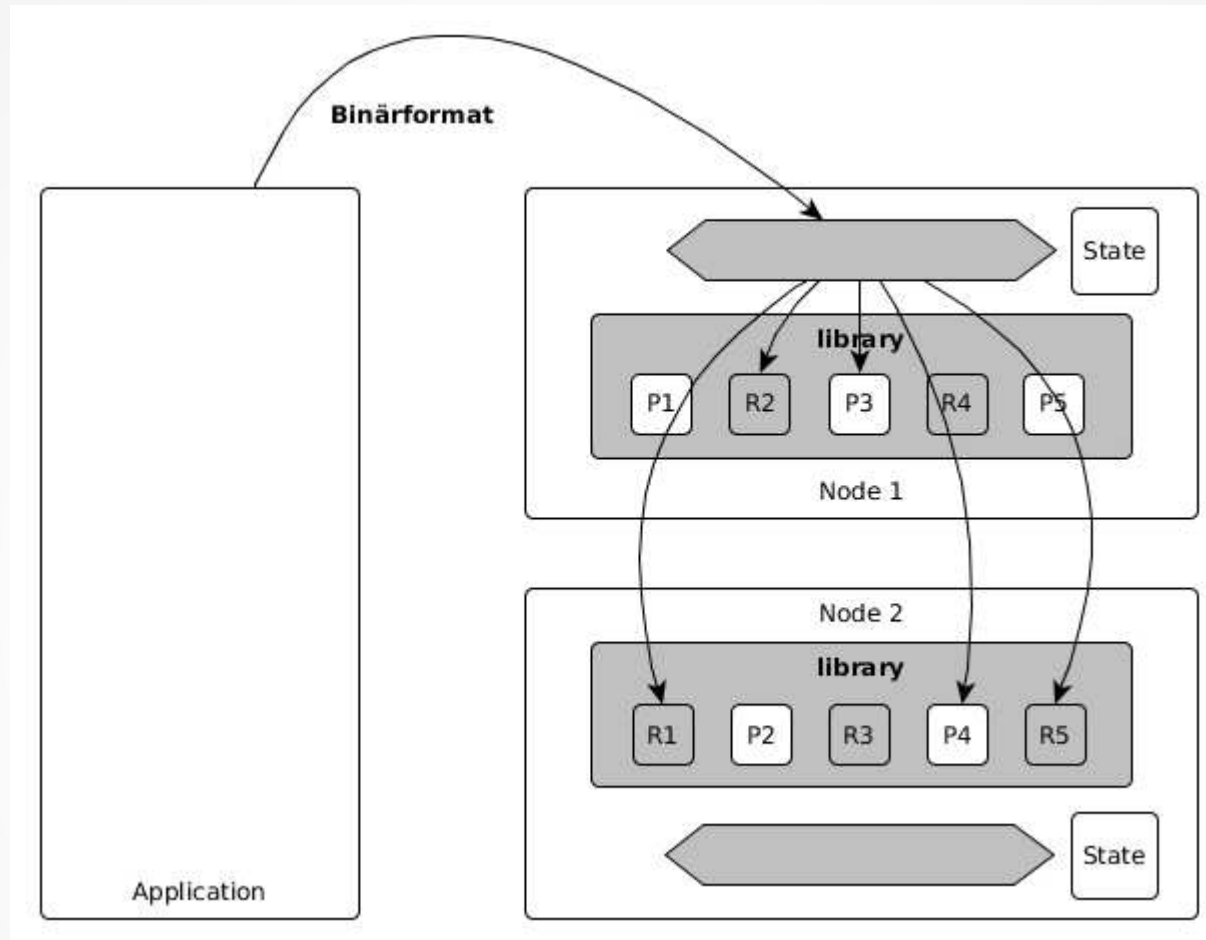


# Transport-Client

- Verbindet sich mit bestehendem Cluster

```
TransportAddress address =  
    new InetSocketAddress("localhost", 9300);  
  
Client client = new TransportClient().  
    addTransportAddress(address);
```

# Transport-Client



# Node-Client

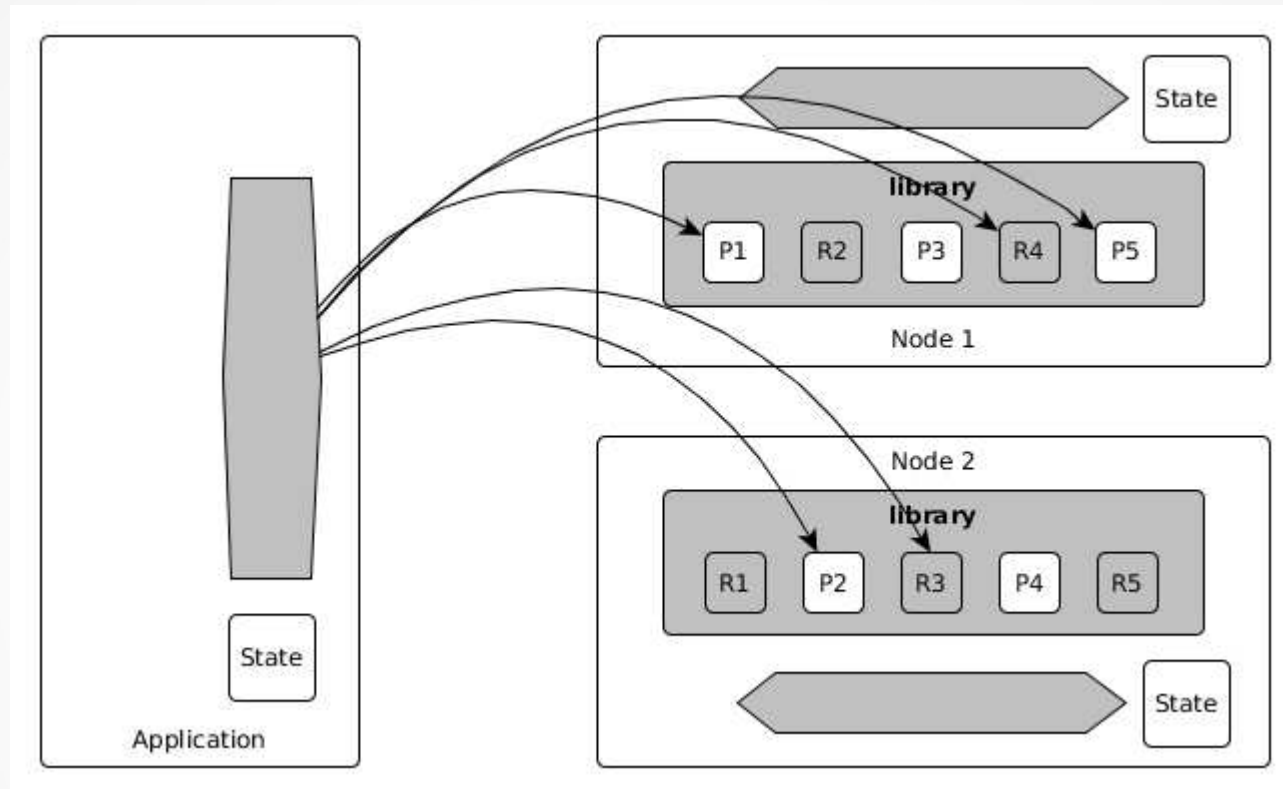
```
Node node = nodeBuilder().client(true).node();  
Client client = node.client();
```

# Node-Client

- Startet eigenen Knoten
- Anwendung wird Teil des Clusters

```
Node node = nodeBuilder().client(true).node();  
Client client = node.client();
```

# Node-Client



# Standard-Client

- Volle API-Unterstützung
- Effiziente Kommunikation
- Node-Client
  - Cluster-State spart unnötige Hops

# Standard-Client Gotchas

- Elasticsearch-Version Client == Server
- Node-Client
  - Bei Start und Stop wird Cluster-State übertragen
- Speicherverbrauch
- Elasticsearch-Dependency

# Alternativen

- Jest: Http-Client
- Spring Data Elasticsearch
- Große Auswahl für JVM-Sprachen

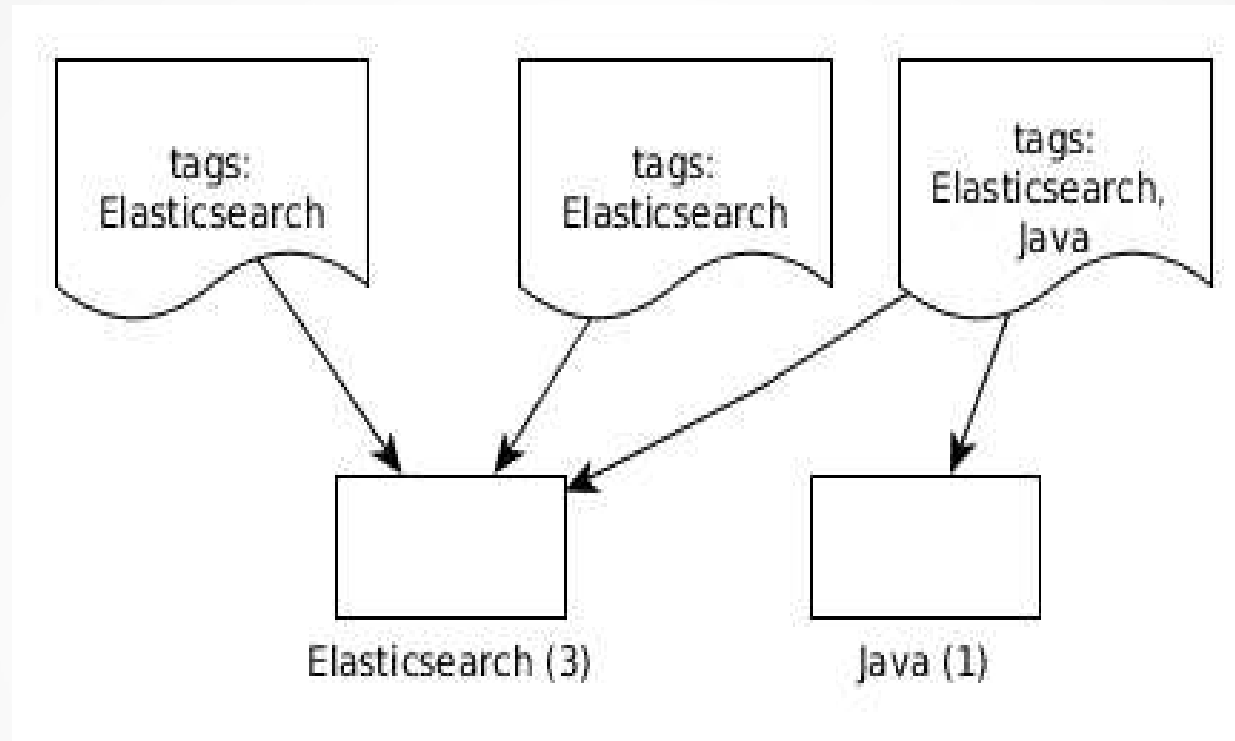


# Aggregationen

# Aggregations

- Informationen über die Daten
- Ersetzen und ermöglichen Facetten
- Anwendungsfälle für Suchanwendungen und Analytics

# Aggregations



# Invertierter Index

Term	Document Id
Elasticsearch	1,2,3
Java	3
Lucene	2,3

# Terms-Aggregation

```
POST /library/book/_search
{
  "size": 0,
  "aggs": {
    "common-tags": {
      "terms": {
        "field": "tags"
      }
    }
  }
}
```

# Terms-Aggregation

```
"aggregations": {  
  "common-tags": {  
    "doc_count_error_upper_bound": 0,  
    "sum_other_doc_count": 0,  
    "buckets": [  
      {  
        "key": "Elasticsearch",  
        "doc_count": 2  
      },  
      {  
        "key": "Lucene",  
        "doc_count": 2  
      },  
      {  
        "key": "Java",  
        "doc_count": 1  
      }  
    ]  
  }  
}
```

[...]

# Terms-Aggregation

- Bucket mit Wert und Anzahl
- Facettierung
- Informationen aus Daten

# Aggregationen kombinieren

```
POST /devoxx/tweet/_search
{
  "aggs" : {
    "hashtags" : {
      "terms" : {
        "field" : "hashtag.text"
      }
    }
  }
}
```



# Aggregationen kombinieren

```
"aggregations": {  
  "hashtags": {  
    "buckets": [  
      {  
        "key": "dartlang",  
        "doc_count": 229  
      },  
      {  
        "key": "java",  
        "doc_count": 216  
      },  
      [...]  
    ]  
  }  
}
```

# Aggregationen kombinieren

```
POST /devoxx/tweet/_search
{
  "aggs" : {
    "hashtags" : {
      "terms" : {
        "field" : "hashtag.text"
      },
      "aggs" : {
        "hashtagusers" : {
          "terms" : {
            "field" : "user.screen_name"
          }
        }
      }
    }
  }
}
```

# Aggregationen kombinieren

```
"key": "scala",
"doc_count": 130,
"hashtagusers": {
  "buckets": [
    {
      "key": "jaceklaskowski",
      "doc_count": 74
    },
    {
      "key": "ManningBooks",
      "doc_count": 3
    },
    [...]
  ]
}
```

# Bucket-Aggregationen

- Range-Aggregationen
- Histogramme
- Filter
- Geo-Aggregationen
- ...

# Metric-Aggregationen

- Berechnen einen oder mehrere Werte
- Meist auf numerischen Feldern
- Stats, Percentiles, Min, Max, Sum, Avg, ...

# Stats-Aggregationen

```
GET /library/book/_search
{
  "aggs": {
    "published_stats": {
      "stats": {
        "field": "published"
      }
    }
  }
}
```

# Stats-Aggregationen

```
"aggregations": {  
  "published_stats": {  
    "count": 5,  
    "min": 14192928000000,  
    "max": 14459904000000,  
    "avg": 14405472000000,  
    "sum": 72027360000000,  
    "min_as_string": "2014-12-23T00:00:00.000Z",  
    "max_as_string": "2015-10-28T00:00:00.000Z",  
    "avg_as_string": "2015-08-26T00:00:00.000Z",  
    "sum_as_string": "2198-03-31T00:00:00.000Z"  
  }  
}
```

# Significant Terms Aggregation

- Findet Besonderheiten in Daten
- Bewertet Datensatz gegen Hintergrundmenge
- Beispiel: Häufige Hashtag-Kombinationen



# Significant Terms Aggregation

```
POST /conf/_search
{
  "aggs": {
    "hashtag": {
      "terms": {
        "field": "hashtag.text"
      },
      "aggs": {
        "associated_hashtags": {
          "significant_terms": {
            "field": "hashtag.text"
          }
        }
      }
    }
  }
}
```

# Significant Terms Aggregation

- bbuzz (Berlin Buzzwords)
  - elasticsearch, devops, bigdata, cassandra
- javaland
  - java8, javafx, javaee7, nighthacking
- dwx14 (Developer Week)
  - newww, nodejs, intelandroid, web, microsoft

# Significant Terms Aggregation

- Empfehlungen
- Fraud-Detection
- Geographische Häufungen

# Recap

- Aggregationen bieten unterschiedliche Einblicke in die Daten
- Facettierung
- Kombination mehrerer Aggregationen
- Grundlage für Visualisierungen

# Zentralisiertes Logging

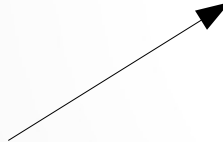
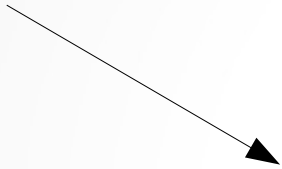
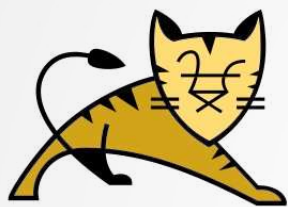
# Logfile-Analyse

- Zentralisierung Logs aus Anwendungen
- Zentralisierung Logs über Maschinen
  - Auch ohne Zugriff
- Leichte Durchsuchbarkeit
- Real-Time-Analysis / Visualisierung
- Daten für alle!

# Logfile-Analyse

- Einlesen
  - Logstash
- Speicherung
  - Elasticsearch
- Auswertung
  - Kibana

# Logfile-Analyse



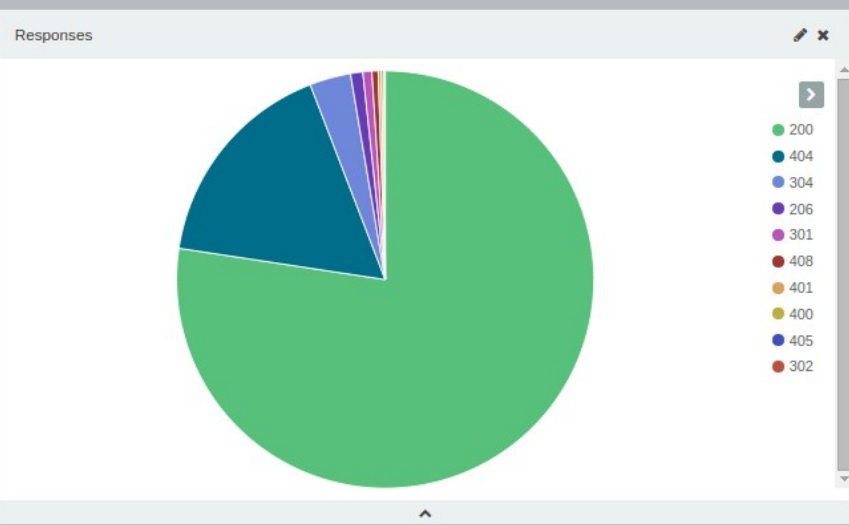
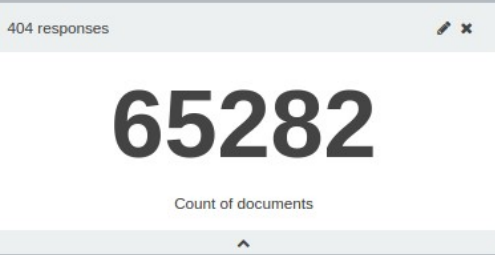
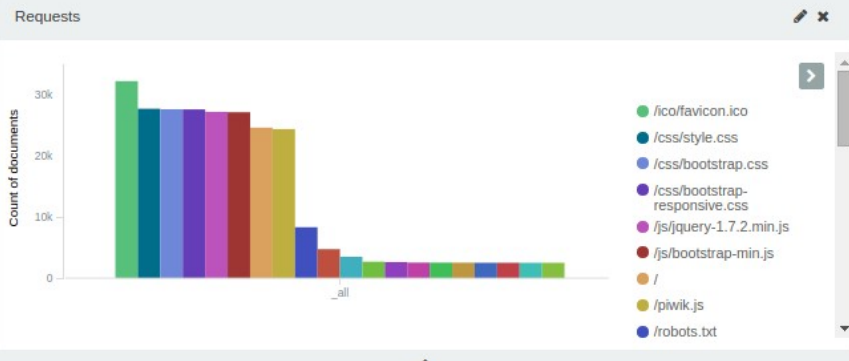


# Logstash-Config

```
input {  
  file {  
    path => "/var/log/apache2/access.log"  
  }  
}  
  
filter {  
  grok {  
    match => { message => "%{COMBINEDAPACHELOG}" }  
  }  
}  
  
output {  
  elasticsearch_http {  
    host => "localhost"  
  }  
}
```

# Kibana

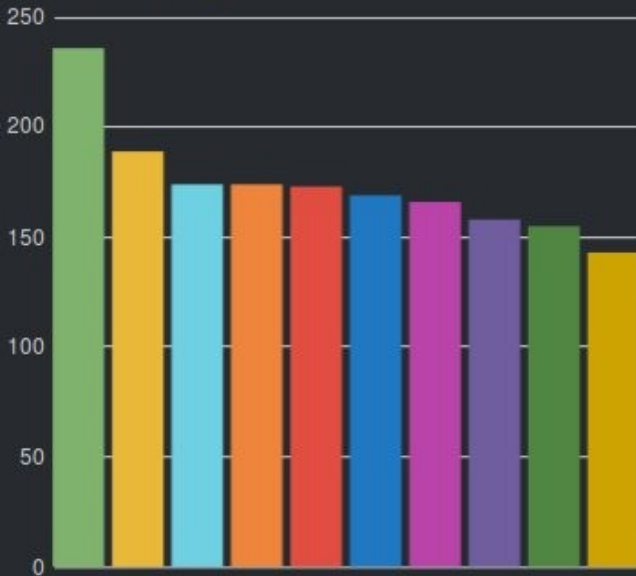
Log-Dashboard



# Kibana

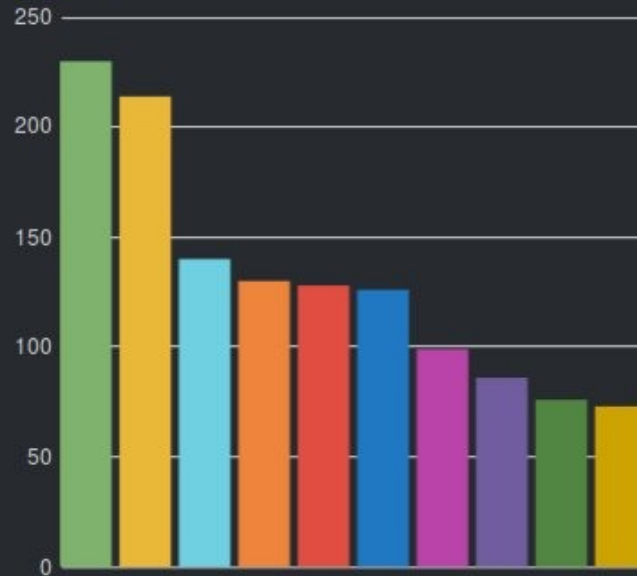
## Mentions

- Stephan007 (235)
- romainguy (173)
- venkat\_s (168)
- Parleys (154)
- chethaase (188)
- BrianGoetz (173)
- arungupta (165)
- jsuereth (142)
- angularjs (172)
- mraible (157)

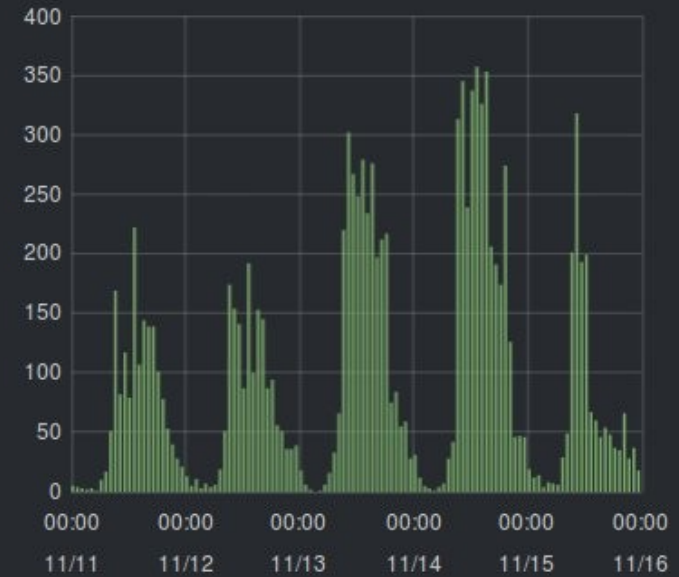


## Hashtags

- dartlang (229)
- scala (129)
- javaavee7 (98)
- dart (72)
- java (213)
- ceylon (127)
- hackergarten (85)
- android (139)
- devoxx4kids (125)
- oracle (75)



Zoom Out | ● (11248) count per 1h | (11248 hits)



# Recap

- Einlesen, Anreichern, Speichern von Logevents
- Zahlreiche Inputs in Logstash
- Konsolidierung
- Zentralisierung
- Auswertung

# Noch viel mehr!

- Unterschiedliche Suchfeatures
- Viele Aggregationen
- Geodaten
- Percolator
- Elasticsearch 2.0

# Weitere Infos



# Weitere Infos

- <http://elastic.co>
- Elasticsearch – The definitive Guide
  - <https://www.elastic.co/guide/en/elasticsearch/guide/master/index.html>
- Elasticsearch in Action
  - <https://www.manning.com/books/elasticsearch-in-action>
- <http://blog.florian-hopf.de>