

# Bleve

Full-Text Indexing and Search for Go  
10 July 2015

Marty Schoch



## Why?

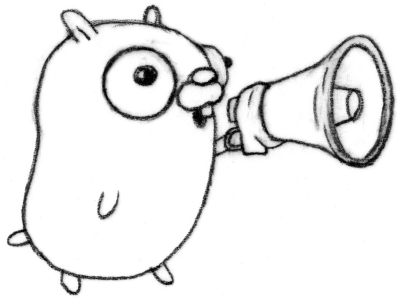
elasticsearch.

*Lucene*



- Lucene/Solr/Elasticsearch are awesome
- Could we build 50% of Lucene's text analysis, combine it with off-the-shelf KV stores and get something interesting?

## Install bleve



*go get [github.com/blevesearch/bleve/...](https://github.com/blevesearch/bleve/)*

# Import

```
10 import "github.com/blevesearch/bleve"  
11  
12 type Person struct {  
13     Name string  
14 }  
15  
16 func main() {  
17     mapping := bleve.NewIndexMapping()  
18     index, err := bleve.New("people.bleve", mapping)  
19     if err != nil {  
20         log.Fatal(err)  
21     }  
22  
23     person := Person{"Marty Schoch"}  
24     err = index.Index("m1", person)  
25     if err != nil {  
26         log.Fatal(err)  
27     }  
28     fmt.Println("Indexed Document")  
29 }
```

# Data Model

```
10 import "github.com/blevesearch/bleve"
11
12 type Person struct {
13     Name string
14 }
15
16 func main() {
17     mapping := bleve.NewIndexMapping()
18     index, err := bleve.New("people.bleve", mapping)
19     if err != nil {
20         log.Fatal(err)
21     }
22
23     person := Person{"Marty Schoch"}
24     err = index.Index("m1", person)
25     if err != nil {
26         log.Fatal(err)
27     }
28     fmt.Println("Indexed Document")
29 }
```

# Index Mapping

```
10 import "github.com/blevesearch/bleve"
11
12 type Person struct {
13     Name string
14 }
15
16 func main() {
17     mapping := bleve.NewIndexMapping()
18     index, err := bleve.New("people.bleve", mapping)
19     if err != nil {
20         log.Fatal(err)
21     }
22
23     person := Person{"Marty Schoch"}
24     err = index.Index("m1", person)
25     if err != nil {
26         log.Fatal(err)
27     }
28     fmt.Println("Indexed Document")
29 }
```

## Create a New Index

```
10 import "github.com/blevesearch/bleve"
11
12 type Person struct {
13     Name string
14 }
15
16 func main() {
17     mapping := bleve.NewIndexMapping()
18     index, err := bleve.New("people.bleve", mapping)
19     if err != nil {
20         log.Fatal(err)
21     }
22
23     person := Person{"Marty Schoch"}
24     err = index.Index("m1", person)
25
26     if err != nil {
27         log.Fatal(err)
28     }
29     fmt.Println("Indexed Document")
30 }
```

# Index Data

```
10 import "github.com/blevesearch/bleve"
11
12 type Person struct {
13     Name string
14 }
15
16 func main() {
17     mapping := bleve.NewIndexMapping()
18     index, err := bleve.New("people.bleve", mapping)
19     if err != nil {
20         log.Fatal(err)
21     }
22
23     person := Person{"Marty Schoch"}
24
25     err = index.Index("m1", person)
26     if err != nil {
27         log.Fatal(err)
28     }
29     fmt.Println("Indexed Document")
30 }
```

Run



# Open Index

```
15 func main() {
16     index, err := bleve.Open("people.bleve")
17     if err != nil {
18         log.Fatal(err)
19     }
20
21     query := bleve.NewTermQuery("marty")
22     request := bleve.NewSearchRequest(query)
23     result, err := index.Search(request)
24     if err != nil {
25         log.Fatal(err)
26     }
27     fmt.Println(result)
28 }
```

# Build Query

```
15 func main() {
16     index, err := bleve.Open("people.bleve")
17     if err != nil {
18         log.Fatal(err)
19     }
20
21     query := bleve.NewTermQuery("marty")
22     request := bleve.NewSearchRequest(query)
23     result, err := index.Search(request)
24     if err != nil {
25         log.Fatal(err)
26     }
27     fmt.Println(result)
28 }
```

# Build Request

```
15 func main() {
16     index, err := bleve.Open("people.bleve")
17     if err != nil {
18         log.Fatal(err)
19     }
20
21     query := bleve.NewTermQuery("marty")
22     request := bleve.NewSearchRequest(query)
23     result, err := index.Search(request)
24     if err != nil {
25         log.Fatal(err)
26     }
27     fmt.Println(result)
28 }
```

# Search

```
15 func main() {
16     index, err := bleve.Open("people.bleve")
17     if err != nil {
18         log.Fatal(err)
19     }
20
21     query := bleve.NewTermQuery("marty")
22     request := bleve.NewSearchRequest(query)
23     result, err := index.Search(request)
24     if err != nil {
25         log.Fatal(err)
26     }
27     fmt.Println(result)
28 }
```

Run

# Query Strings

Simple query language for humans

```
+content:debugger word_count:>30
```

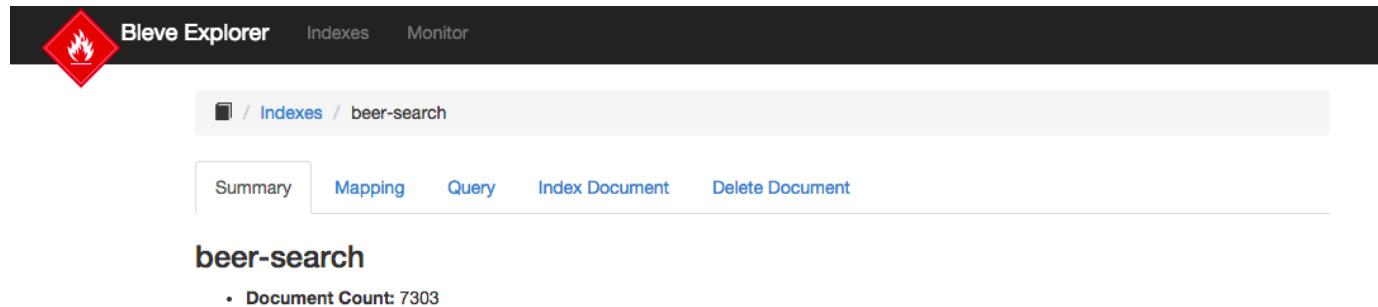
```
title:"delve debugger" title:go~2 -content:rust
```

# Optional HTTP Handlers

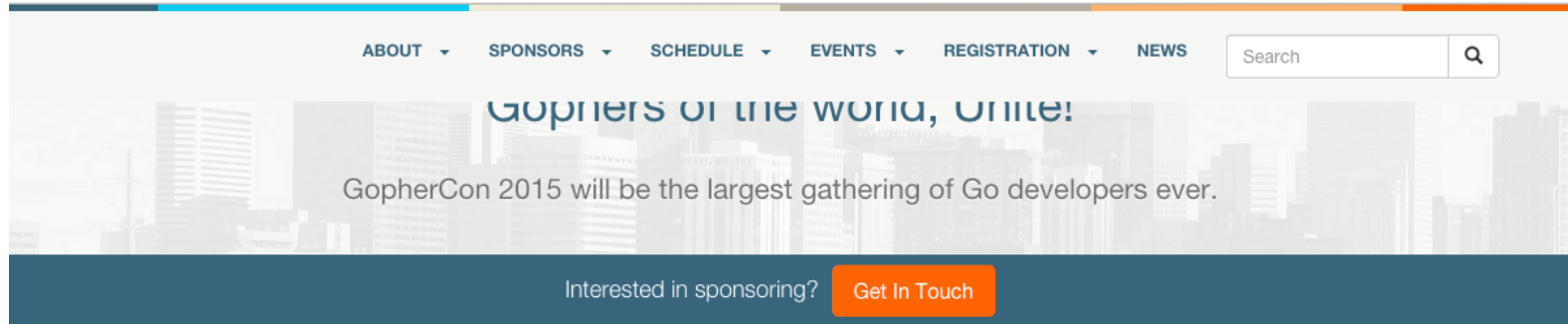
```
import "github.com/blevesearch/bleve/http"
```

- All major bleve operations mapped
- Assume JSON document bodies
- See bleve-explorer sample app

```
https://github.com/blevesearch/bleve-explorer
```



# GopherCon Site Search



denver

2 results (6ms)

June 7th

3.08

**Brian Ketelsen** on 2015-04-14T00:00:00Z

...herCon Kickoff Party 6:00pm - 10:00pm - Galvanize, 1062 Delaware Street, Denver, CO Join us for the Pre-Party and kick off GopherCon 2015 in style! More details available on the Denver Gophers mee...

July 8th

3.02

**Brian Ketelsen** on 2015-04-14T00:00:00Z

...pm Location: Wynkoop Brewing Company 1634 18th St. Denver, CO 80202 ...

## Refine Results

### Modified

30+ days (2)

### Types

schedule (2)

## Join the Community

The logo for freenode, featuring the word "freenode" in a lowercase, sans-serif font. The "free" part is in white and the "node" part is in a bright green color. The text is set against a dark grey rectangular background.

- #bleve is small/quiet room, talk to us real time

The logo for Google Groups, featuring the word "Google" in its multi-colored font followed by the word "groups" in a blue, lowercase, sans-serif font.

- Discuss your use-case
- Plan a feature implementation

# GitHub

- Apache License v2.0, Report Issues, Submit Pull Requests



# Thank you

Marty Schoch

[marty@couchbase.com](mailto:marty@couchbase.com) (<mailto:marty@couchbase.com>)

<http://github.com/blevesearch/bleve> (<http://github.com/blevesearch/bleve>)

<http://www.blevesearch.com/> (<http://www.blevesearch.com/>)

[@mschoch](http://twitter.com/mschoch) (<http://twitter.com/mschoch>)

[@blevesearch](http://twitter.com/blevesearch) (<http://twitter.com/blevesearch>)

