CouchDB — Databáze pro web

Karel Minařík

Karel Minařík

→ Web designer a vývojář na volné noze od roku 2000

→ V minulosti Flash vývojář, art director, informační architekt v internetové agentuře, „..více na LinkedIn

→ Ruby, Rails, Git, CouchDB *propagandista* v České Republice

→ **karmi.cz**

CouchDB

relax

Apache CouchDB is a distributed, fault-tolerant and schema-free document-oriented database accessible via a RESTful HTTP/JSON API.

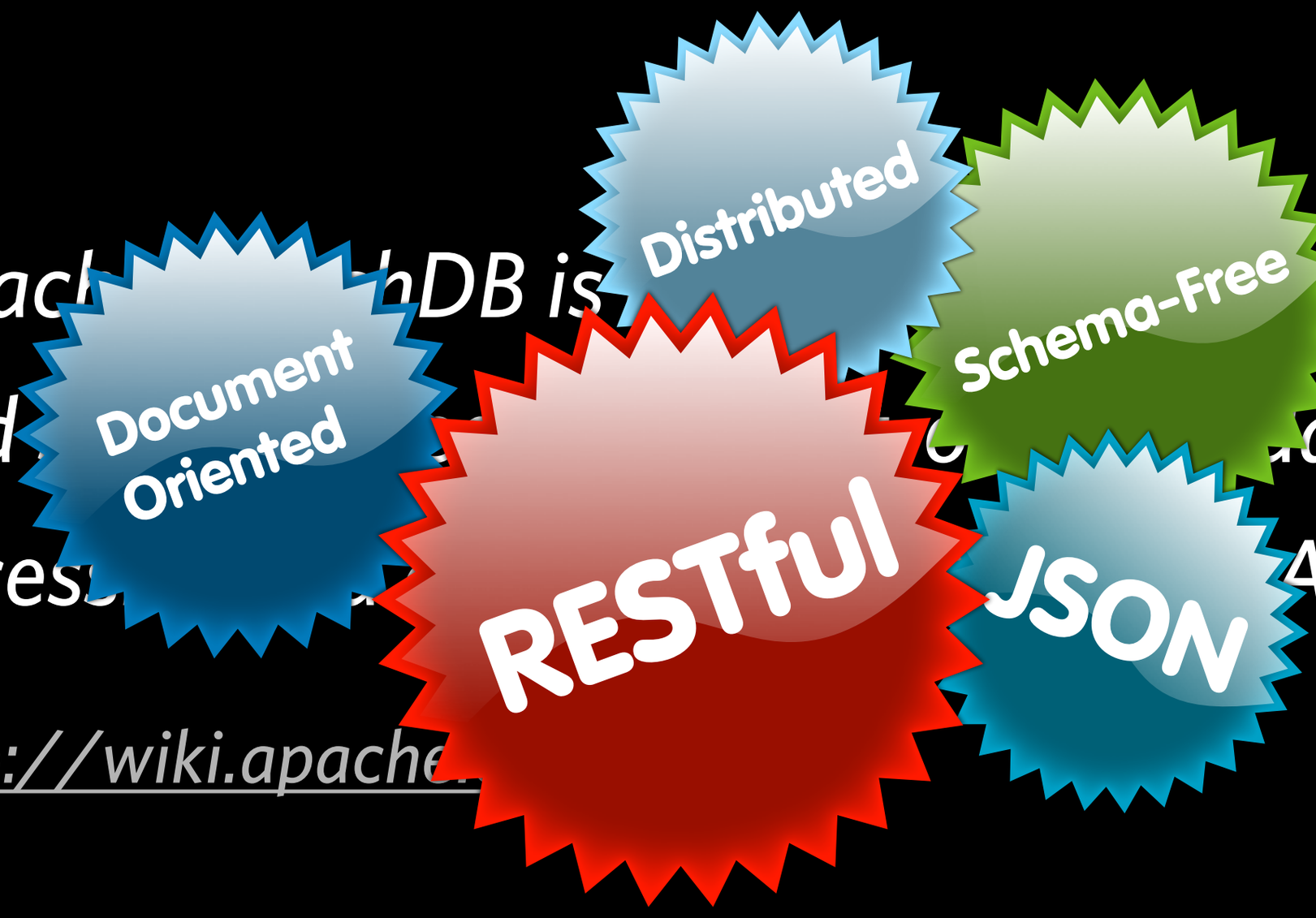
http://wiki.apache.org/couchdb

Apac hDB is -tolerant

and atabase

acces API.

http://wiki.apache.




NO SQL !!!1
Replication!
Decentralization!
Schema FREE!!!
JSON, not XML!
Clusters!
HTTP!!!!!
FAIL

Project Voldemort

Důvody vzniku a rozšíření „netradičních“ databází

NoSQL není ani „protestní hnutí" „přechodná móda".

Důvody pro vznik a rozšíření nových databází jsou skutečné.

A většinou pocházejí ze skutečné „bolesti“.

Denormalizace databáze v Digg.com

```
SELECT `digdate`, `id` FROM `Diggs`
WHERE `userid` IN (1, 2, 3, 4, ... 1000000)
AND itemid = 123 ORDER BY `digdate` DESC, `id` DESC;
```

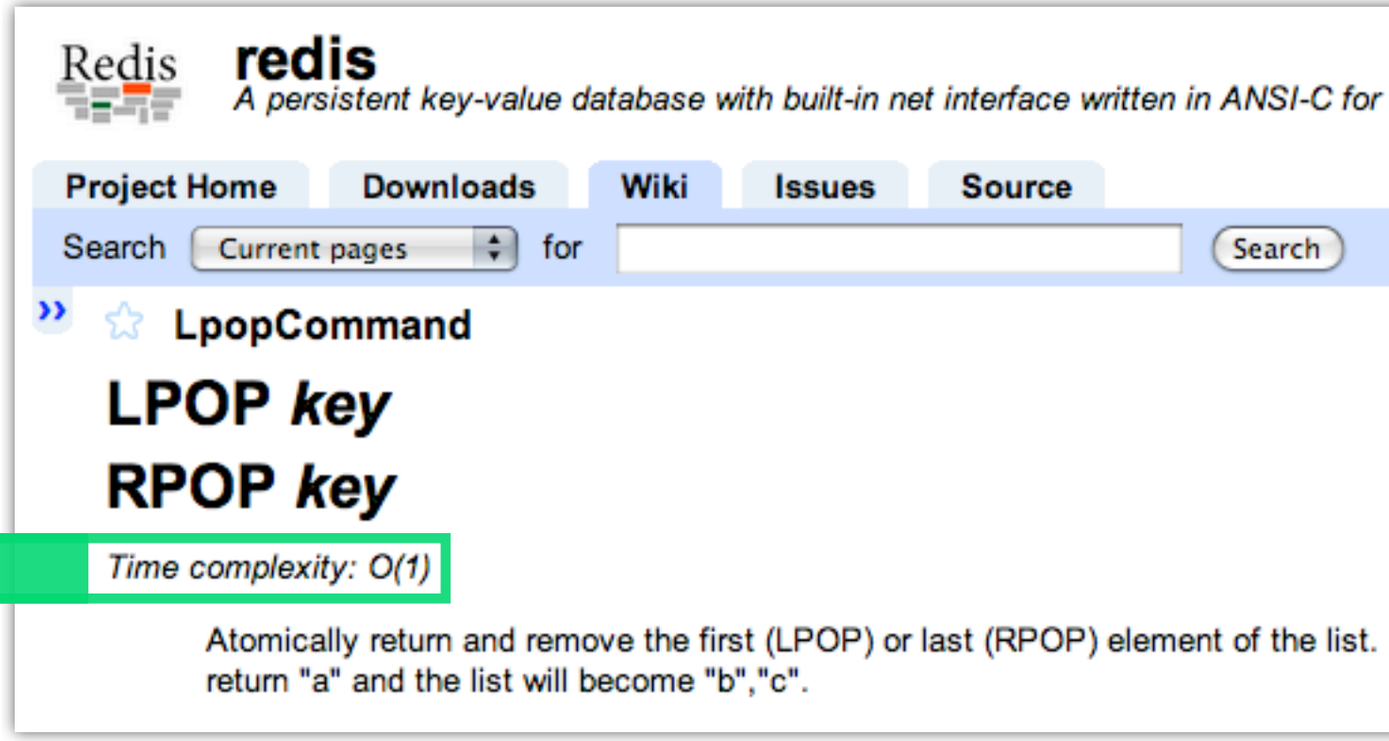
"A full query can actually clock in at 1.5kb, which is many times larger than the actual data we want. With a cold cache, this query can take 14 seconds to execute."

"Non-relational data stores reverse this model completely, because they don't have the complex read operations of SQL. The model forces you to shift your computation to the writes, while reducing most reads to simple operations – the equivalent of `SELECT * FROM `Table`.`"

http://about.digg.com/blog/looking-future-cassandra

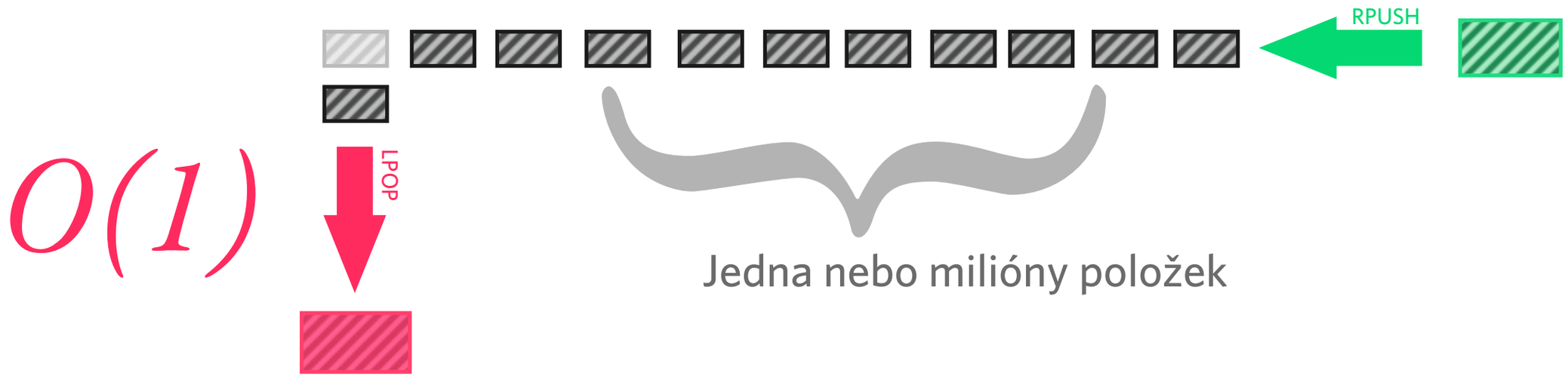
Redis: „asymptotická složitost“ součástí balení

```
redis> rpush mylist 1

redis> rpush mylist 2

redis> lpop mylist

"1"

...

redis> llen mylist

(integer) 1 000 000

redis> lpop mylist

"2"
```




```
$ redis-benchmark
...
====== LPOP ======
10010 requests completed in 0.47 seconds
...
96.22% <= 3 milliseconds
```

Příklad: Asynchronní fronty

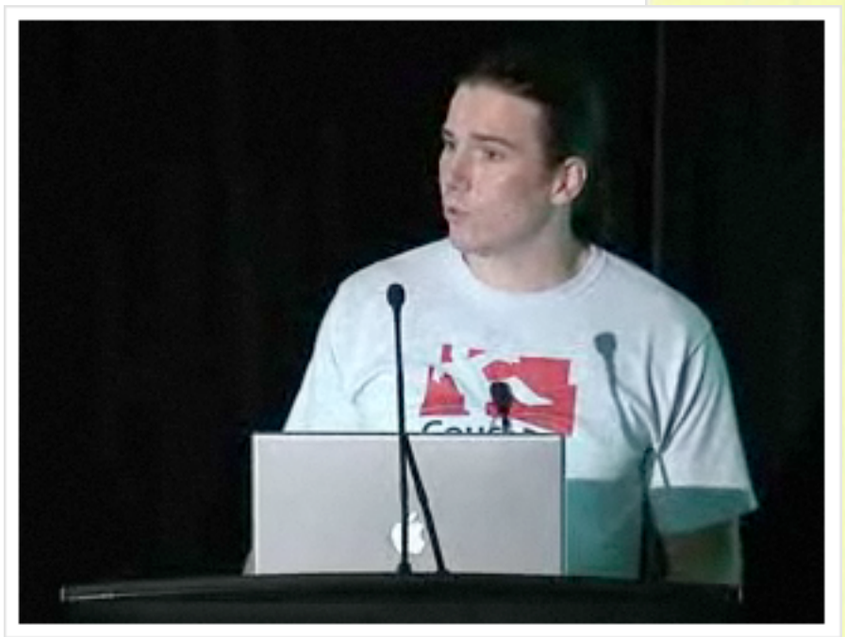


http://github.com/defunkt/resque/blob/master/lib/resque.rb#L133-138

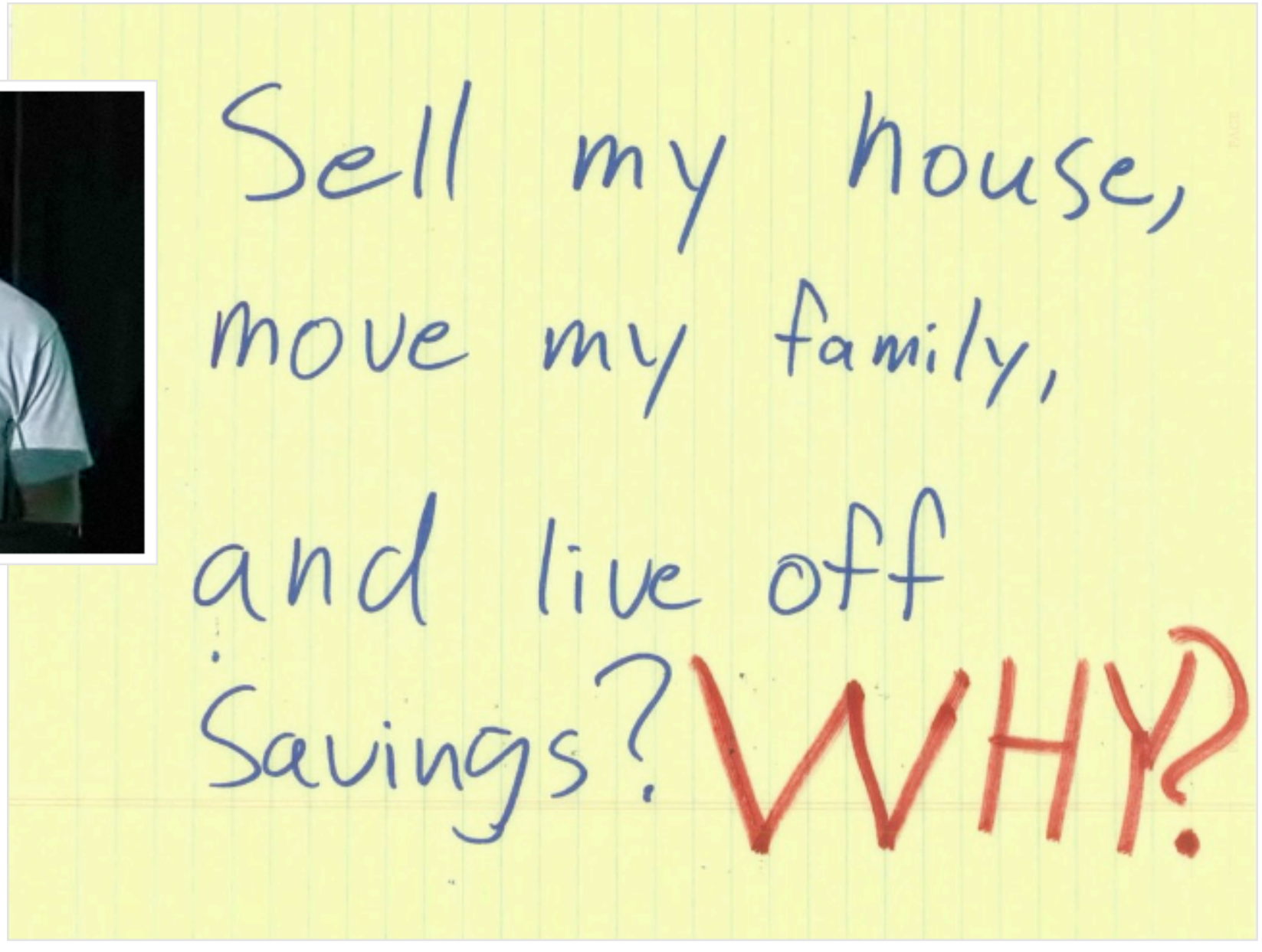
```
133     # Pops a job off a queue. Queue name should be a string.
134     #
135     # Returns a Ruby object.
136     def pop(queue)
137       decode redis.lpop("queue:#{queue}")
138     end
```

1 Příběh CouchDB

Damien Katz: CouchDB and Me



Damien Katz
(RubyFringe 2008)



Damien Katz: CouchDB and Me

Na začátku bylo C++, XML a vlastní dotazovací jazyk.

Zkrátka všechno to, co za ještě nikoho nevyhodili.

Pak přišel Erlang, HTTP, JSON a Map/Reduce.

2 Bezeschémové dokumenty

“Relační data”

To ale **neznamená**, že svět odpovídá **třetí normální formě.**

Učebnicový příklad

Navrhněte **databázi zákazníků.**

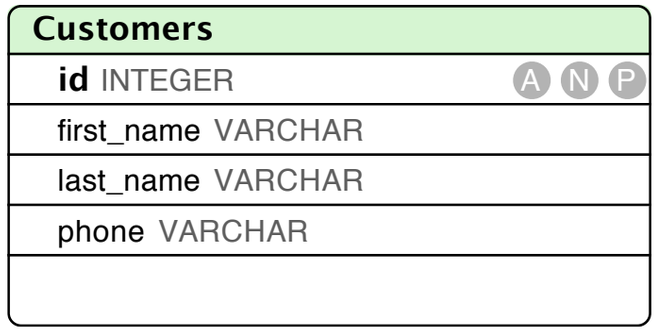
Lidé mají jména, e-maily, telefonní čísla, …

Kolik telefonních čísel?

Učebnicový příklad

Relační databáze, 1. semestr

Customer ID	First Name	Surname	Telephone Number
123	Robert	Ingram	555-861-2025
456	Jane	Wright	555-403-1659
789	Maria	Fernandez	555-808-9633

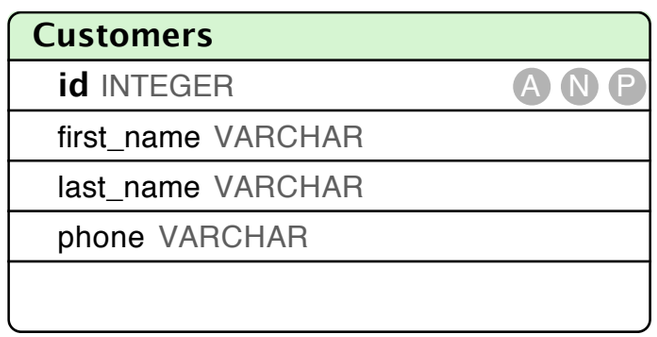


Tak. A co s více telefonními čísly?

Učebnicový příklad

„Řešení", pokus 1

Customer ID	First Name	Surname	Telephone Numbers
123	Robert	Ingram	555-861-2025
456	Jane	Wright	555-403-1659, 555-776-4100
789	Maria	Fernandez	555-808-9633



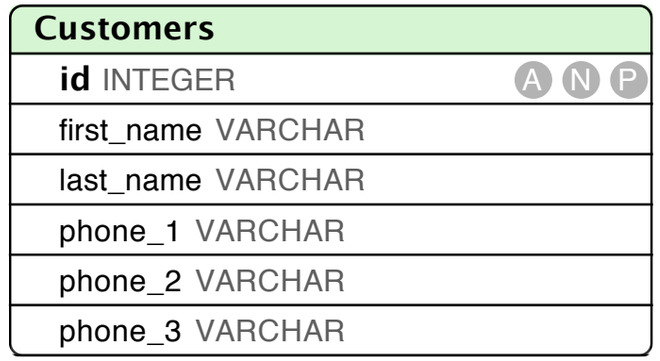
"Stejně budeme s tou databází pracovat jen z naší aplikace."

http://en.wikipedia.org/wiki/First_normal_form#Domains_and_values

Učebnicový příklad

„Řešení", pokus 2

Customer ID	First Name	Surname	Tel. No. 1	Tel. No. 2	Tel. No. 3
123	Robert	Ingram	555-861-2025		
456	Jane	Wright	555-403-1659	555-776-4100	555-403-1659
789	Maria	Fernandez	555-808-9633		



"Tohle je přece o moc lepší *architektura*!"

Dobře. Pak zkuste odpovědět na následující otázky:

- Jak vyhledáte zákazníky podle telefonního čísla?
- Kteří zákazníci mají stejné telefonní číslo?
- Kolik telefonních čísel má zákazník?

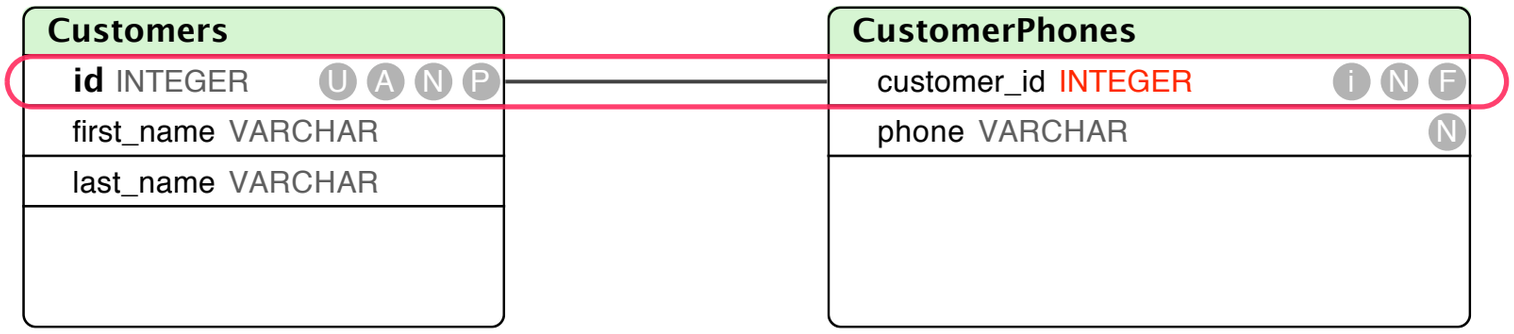
A nyní prosím přidejte možnost ukládat **čtyři** telefonní čísla. Díky.

Učebnicový příklad

Správné řešení

Customer ID	First Name	Surname
123	Robert	Ingram
456	Jane	Wright
789	Maria	Fernandez

Customer ID	Telephone Number
123	555-861-2025
456	555-403-1659
456	555-776-4100
789	555-808-9633



Učebnicový příklad

```
mysql> SELECT * FROM Customers LEFT JOIN CustomerPhones
                              ON Customers.id = CustomerPhones.customer_id;

+----+------------+-----------+-------------+-------+
| id | first_name | last_name | customer_id | phone |
+----+------------+-----------+-------------+-------+
|  1 | John       | Smith     | 1           | 123   |
|  1 | John       | Smith     | 1           | 456   |
+----+------------+-----------+-------------+-------+
```

Učebnicový příklad

```
mysql> SELECT * FROM Customers WHERE id = 1;

+----+------------+-----------+
| id | first_name | last_name |
+----+------------+-----------+
|  1 | John       | Smith     |
+----+------------+-----------+
```

```
mysql> SELECT phone FROM CustomerPhones WHERE customer_id = 1;

+-------+
| phone |
+-------+
| 123   |
| 456   |
+-------+
```

Strukturované dokumenty

Ale hrome!, potřebujeme spíše něco jako:

```
{
  "id"         : 1,

  "first_name" : "Clara",
  "last_name"  : "Rice",

  "phones"     : ["0000 777 888 999", "0000 123 456 789", "0000 314 181 116"]
}
```

“V pohodě. Prostě iteruj přes řádky a sestav si objekt tak, jak ho potřebuješ. Tak to prostě je.“

“Když to bude zlobit, dáme tam nějakou *cache*.”

Efemérní data

Ne všechno ale musí být uděláno „správně“. Nebo musí?

```
class User < ActiveRecord::Base
  serialize :preferences
end
```

„Konzistence“

Může „správné řešení“ také selhat?

Může. A zatraceně snadno.

PŘÍKLAD

Když navrhujete fakturační aplikaci, ukládáte zákazníka k faktuře „správně“, přes cizí klíče.

Pak se změní **adresa zákazníka**.

Změnila se také **adresa na faktuře**?

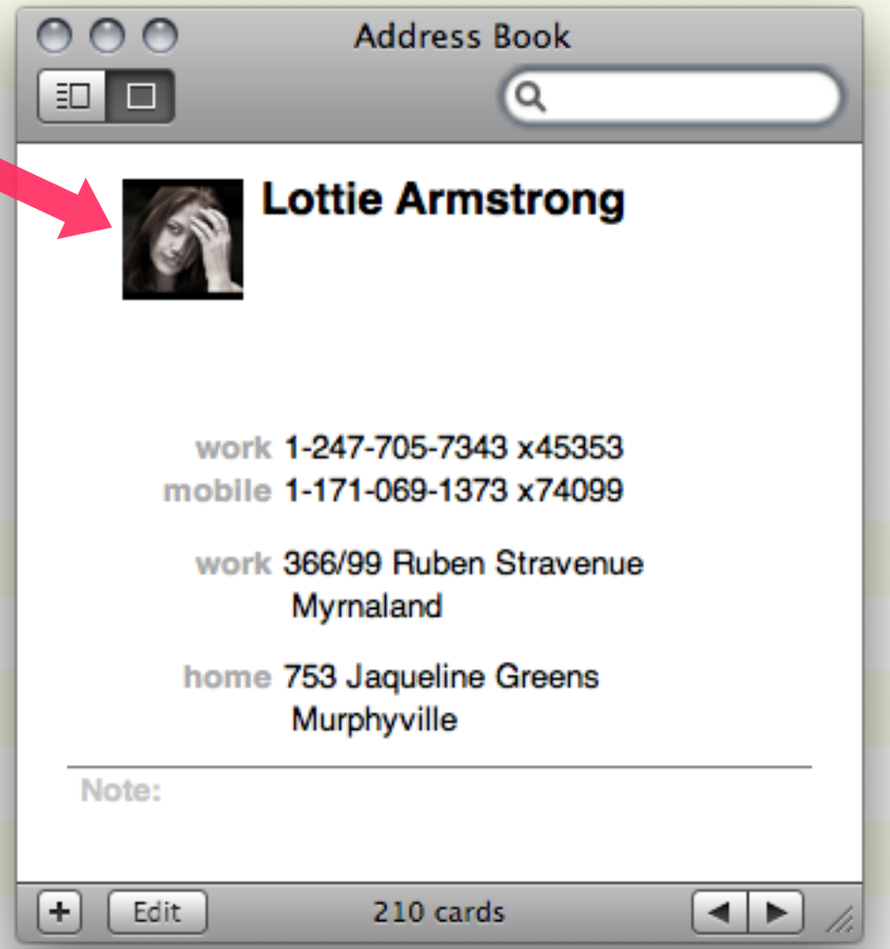
Dokumenty v reálném světě

Dokumenty v reálném světě


```
{
    "_id"          : "clara-rice",
    "_rev"         : "1-def456",

    "first_name"   : "Clara",
    "last_name"    : "Rice",

    "phones"       : {
      "mobile" : "0000 777 888 999",
      "home"   : "0000 123 456 789",
      "work"   : "0000 314 181 116"
    },

    "addresses"    : {
      "home"   : {
        "street"  : "Wintheiser Ports",
        "number"  : "789/23",
        "city"    : "Erinshire",
        "country" : "United Kingdom"
      },
    },

    "occupation" : "model",
    "birthday"   : "1970/05/01",

    "groups"     : ["friends", "models"],

    "created_at" : "2010/01/01 10:00:00 +0000"
}
```

Dokumenty v reálném světě

Field	Value
_id	"lottie-armstrong"
_rev	"2-fcb71b26096957b3ff3ffd2970f3c933"
_attachments	portrait.jpg 72.5 KB, image/jpeg
addresses	home number "753" city "Murphyville" street "Jaqueline Greens" country "United Kingdom" work number "366/99" city "Myrnaland" street "Ruben Stravenue" country "New Zealand"
birthday	"1952/10/24"
first_name	"Lottie"
groups	0 "work"
last_name	"Armstrong"
occupation	"programmer"
phones	cell "1-171-069-1373 x74099" work "1-247-705-7343 x45353"
Showing revision 2 of 2	← Previous Version




Prokrustovo lože

3 RESTful HTTP

PROPRIETE CERN
CERN EUROPEAN ORGANIZATION FOR NUCLEAR
PROPRIETE CERN
Information Management: A Proposal
1990s

Vytvořeno „z webu“

Django may be built for the Web,
but CouchDB is built of the Web.

I've never seen software that so completely embraces the philosophies behind HTTP.

Jacob Kaplan-Moss, *Of the Web* (2007)

http://jacobian.org/writing/of-the-web/

Vytvořeno „z webu“

HTTP is the lingua franca of our age; if you speak HTTP, it opens up all sorts of doors.

There's something almost subversive about CouchDB; it's completely language-, platform-, and OS-agnostic.

http://jacobian.org/writing/of-the-web/

HTTP rozhraní

```
HOST=http://localhost:5984

curl -X GET $HOST
# {"couchdb":"Welcome","version":"0.11.0b22c551bb-git"}

curl -X GET $HOST/my-database
# {"error":"not_found","reason":"no_db_file"}

curl -X PUT $HOST/my-database
# {"ok":true}

curl -X PUT $HOST/my-database/abc123 -d '{"foo":"bar"}'
# {"ok":true,"id":"abc123","rev":"1-4c6114c65e295552ab1019e2b046b10e"}

curl -X GET $HOST/my-database/abc123
# {"_id":"abc123","_rev":"1-4c6114c65e295552ab1019e2b046b10e","foo":"bar"}

curl -X DELETE $HOST/my-database/abc123?rev=2-d179f665eb01834faf192153dc72dcb3
# {"ok":true,"id":"abc123","rev":"1-4c6114c65e295552ab1019e2b046b10e"}
```

Jednoduchá implementace API

```
require 'rubygems'
require 'ostruct'

require 'restclient'
require 'json'
```

1 HTTP knihovna

2 JSON knihovna

```
class Article < OpenStruct

  def self.db(path='')
    RestClient::Resource.new "http://localhost:5984/blog/#{path}",
                             :headers => { :content_type => :json, :accept => :json }
  end

  db.put '' rescue RestClient::PreconditionFailed

  def self.create(params={})
    new db.post(params.to_json)
  end

  def self.find(id)
    new JSON.parse( db(id).get )
  end

  def destroy
    self.class.db(self._id + "?rev=#{self._rev}").delete
  end

end
```

Jednoduchá implementace API

```
Article.create :_id   => 'my-first-post',
               :title => 'CouchDB is easy',
               :body  => 'So relax!',
               :tags  => ['couchdb', 'databases'] rescue RestClient::Conflict

article =  Article.find('my-first-post')

puts "Got an article:"
p article

puts "\n------"
puts "Title: %s" % article.title         + "     (class: #{article.title.class})"
puts "Tags:  %s" % article.tags.inspect + "     (class: #{article.tags. class})"
puts "------\n\n"

puts "Deleting article..."
article.destroy
```

HTTP „od sklepa až na půdu“

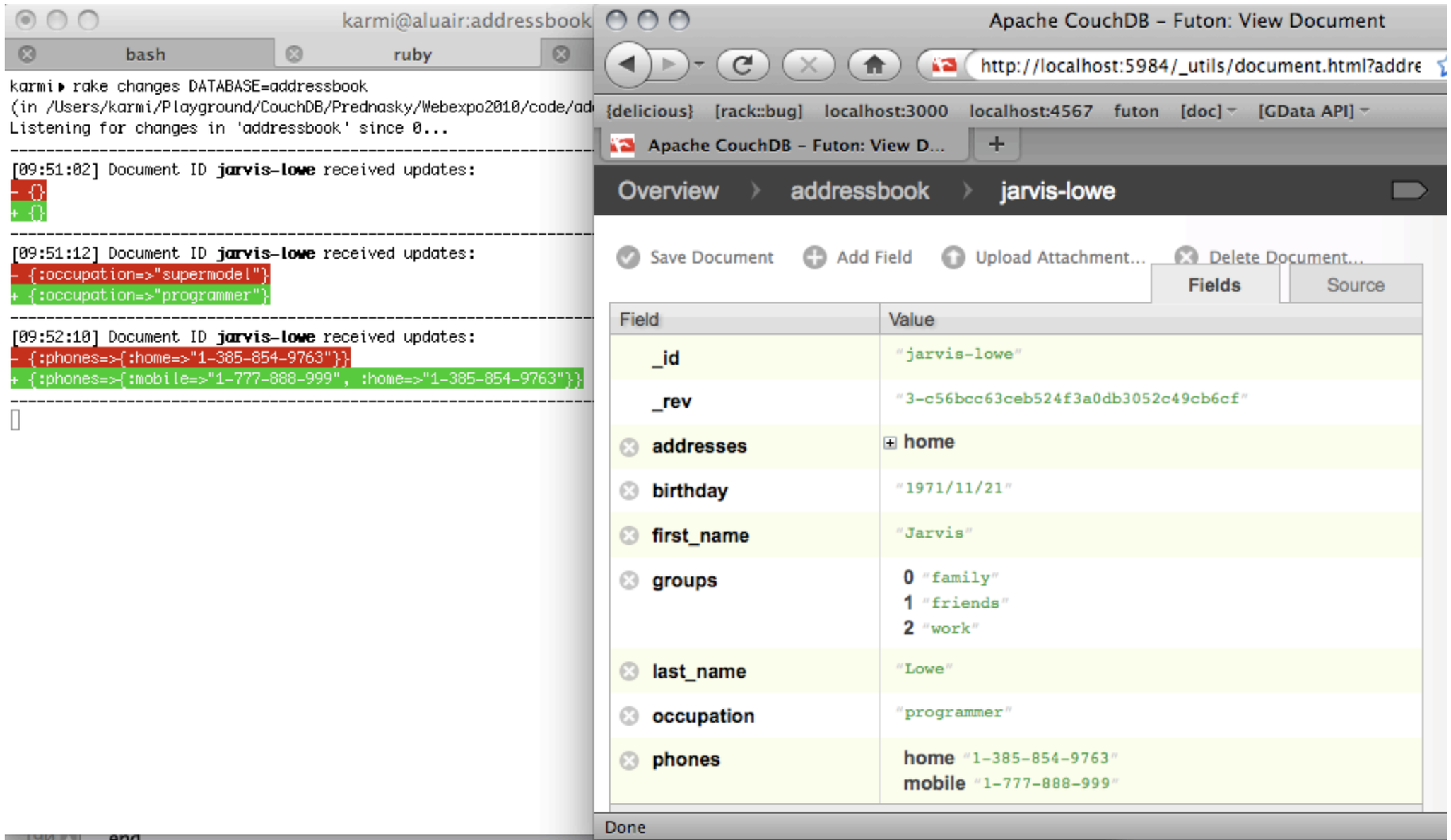
```
$ curl -X POST http://localhost:5984/_replicate \
       -d '{"source":"database",
            "target":"http://example.org/database"}'
```

Skutečné využití HTTP

```
$ curl -i -X GET $HOST/my-database/abc123

HTTP/1.1 200 OK
Server: CouchDB/1.0.1 (Erlang OTP/R14B)
Etag: "4-f04f2435e031054d6b5298c5841ae052"
Date: Thu, 23 Sep 2010 12:56:37 GMT
Content-Type: text/plain;charset=utf-8
Content-Length: 73
Cache-Control: must-revalidate

{"_id":"abc123","_rev":"4-f04f2435e031054d6b5298c5841ae052","foo":"bar"}
```

```
$ cat /etc/squid3/squid.conf

cache_peer 192.168.100.2 parent 5984 0 no-query originserver name=master
acl master_acl method GET
cache_peer_access master allow master_acl
```

Notifikace o změnách (_changes)




```ruby
end

# ---> Run the _changes listener
begin
  CouchDB::Changes.new(ENV['DATABASE']).listen
rescue ArgumentError => e
  puts "[!] #{e.message}"
  exit(1)
end
```

4 Spolehlivost a paralelizace

CouchDB nemá vypínač.

CouchDB nemá příkaz *repair*.

```
$ kill -9 <PID>
```

Erlang

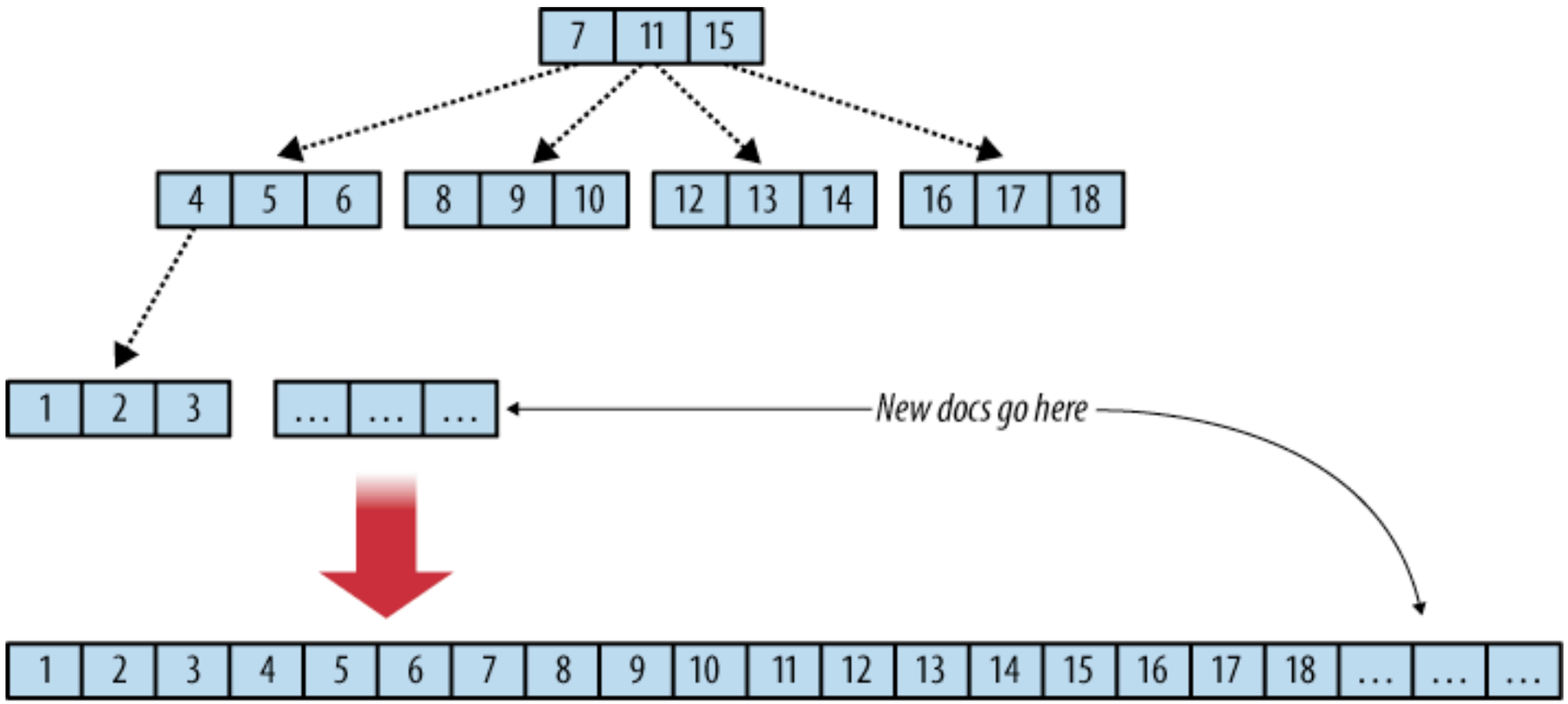
http://www.youtube.com/watch?v=uKfKtXYLG78

Erlang

Erlang's main strength is support for concurrency. It has a small but powerful set of primitives to create processes and communicate among them.
(…) a benchmark with 20 million processes has been successfully performed.

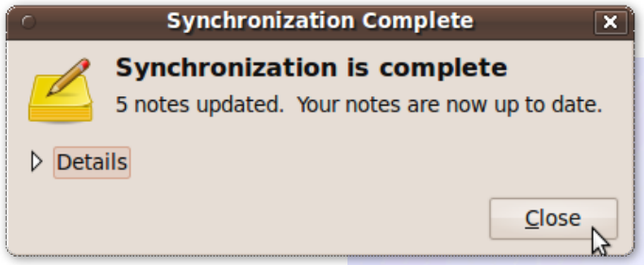
http://en.wikipedia.org/wiki/Erlang_(programming_language)

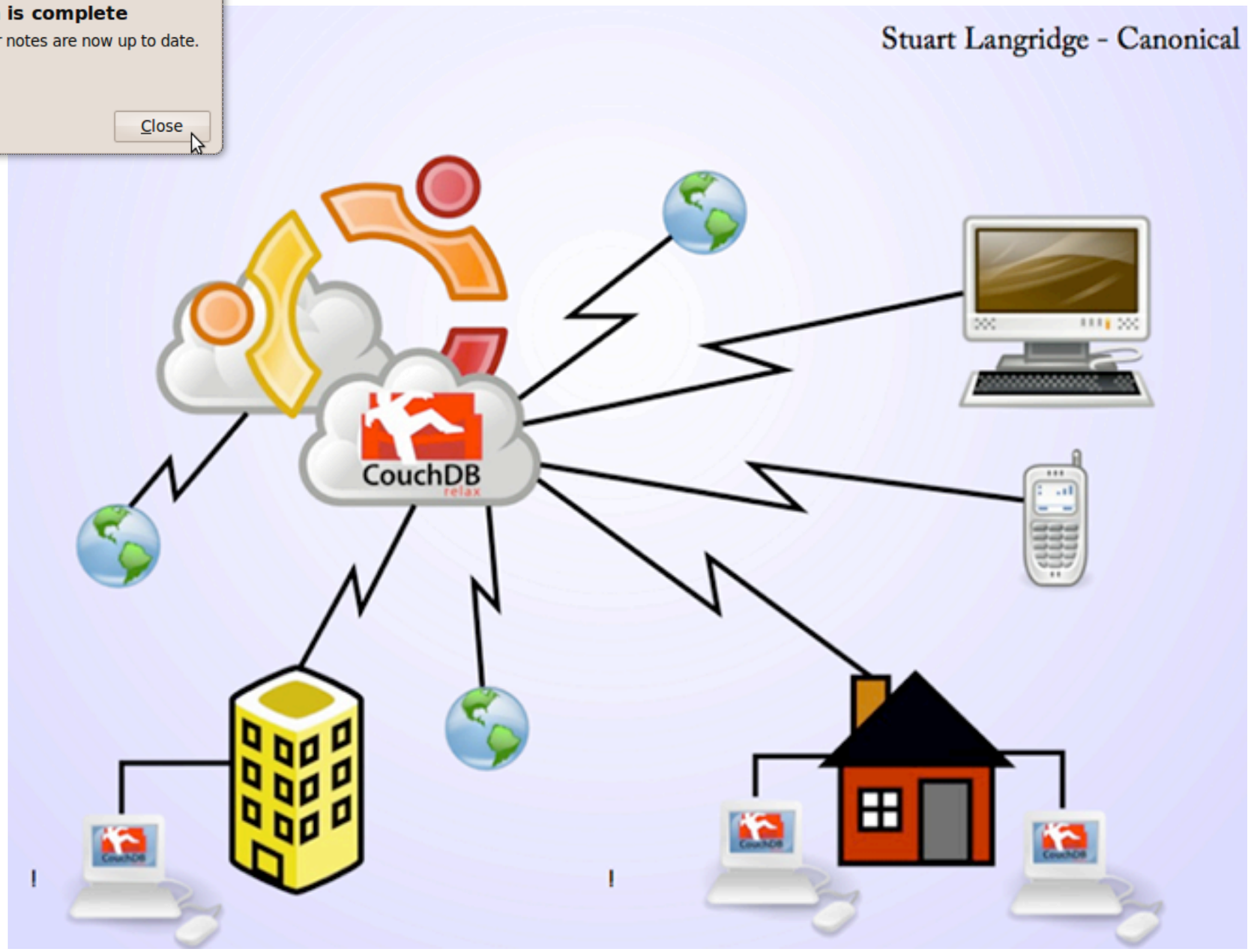
„Append-Only B-Tree“



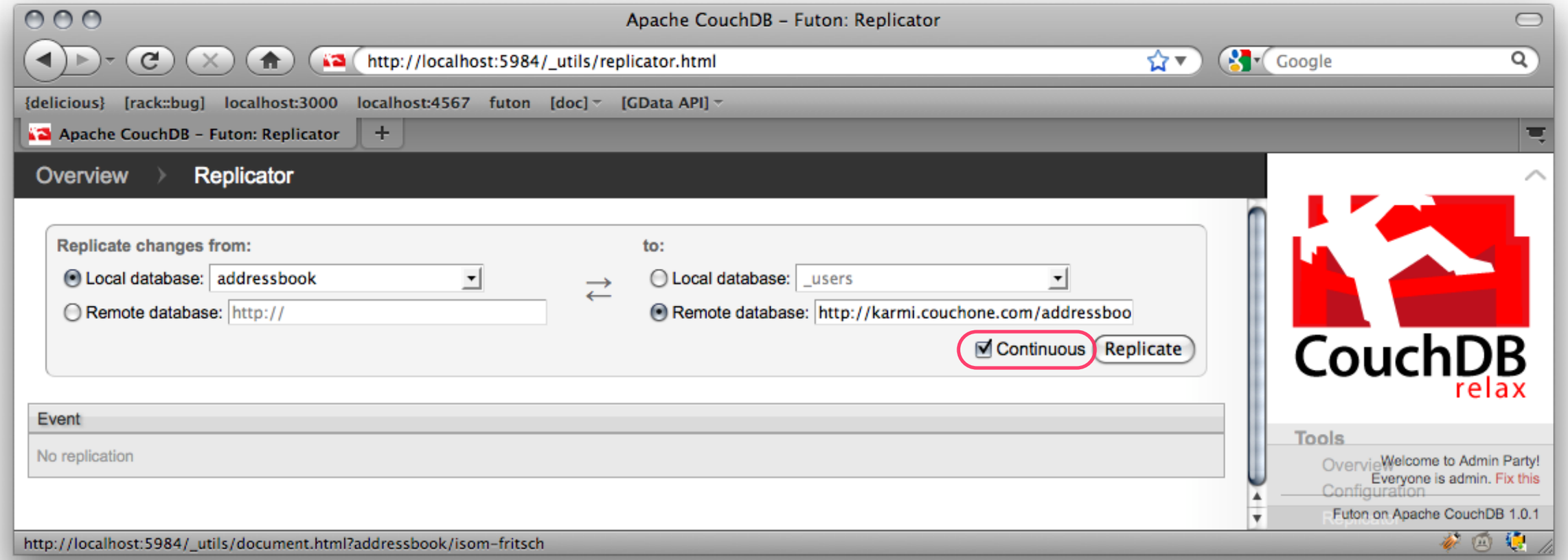
Decentralizace

Ubuntu One

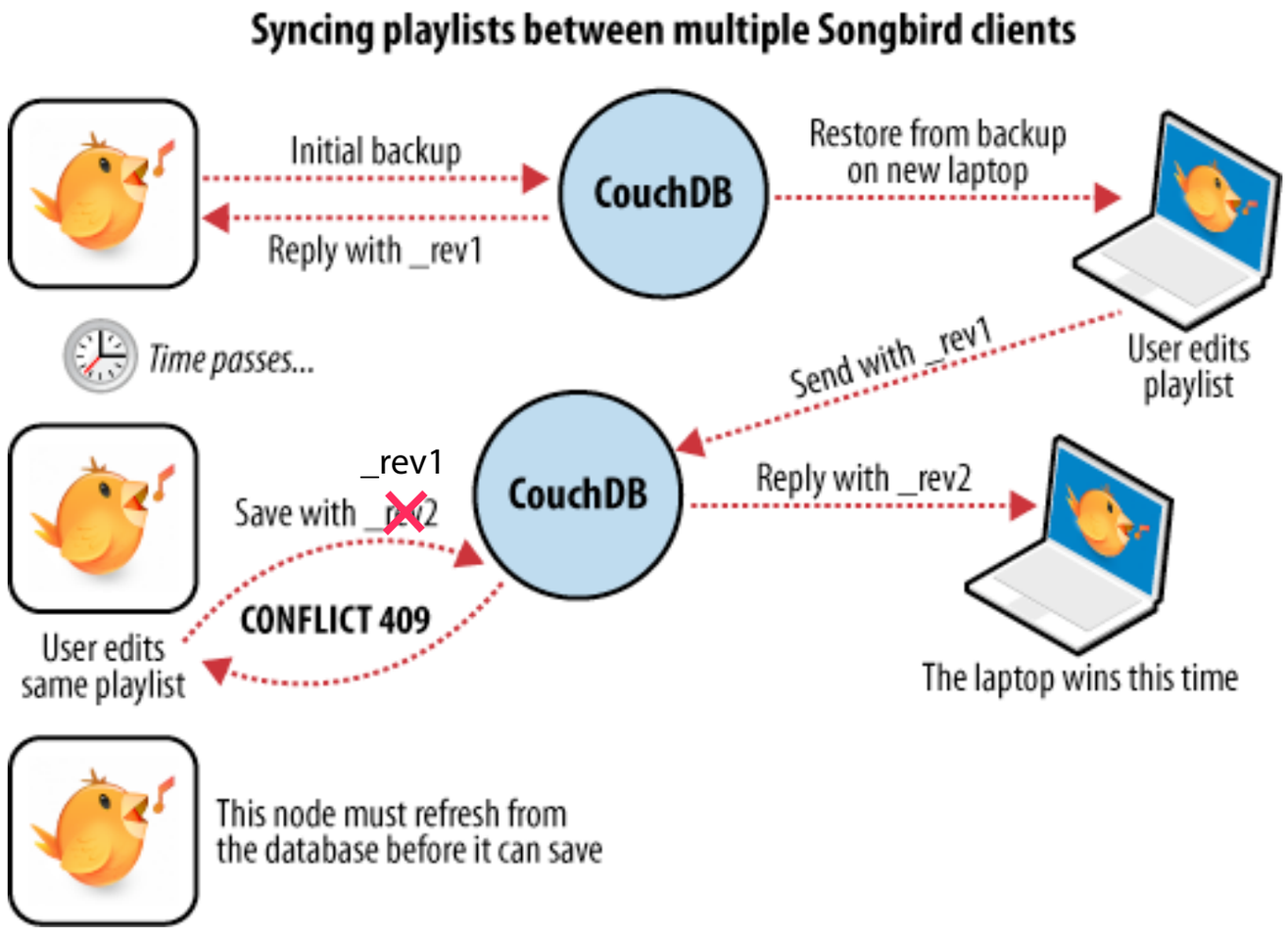

Synchronization Complete
Synchronization is complete
5 notes updated. Your notes are now up to date.
Details
Close


Stuart Langridge - Canonical
CouchDB
relax

Replikace


Apache CouchDB – Futon: Replicator
http://localhost:5984/_utils/replicator.html
Google
{delicious} [rack::bug] localhost:3000 localhost:4567 futon [doc] [GData API]
Apache CouchDB – Futon: Replicator
Overview › Replicator
Replicate changes from:
to:
Local database: addressbook
Remote database: http://
Local database: _users
Remote database: http://karmi.couchone.com/addressboo
Continuous
Replicate
Event
No replication
CouchDB
relax
Tools
Overview
Configuration
Replicator
Welcome to Admin Party!
Everyone is admin. Fix this
Futon on Apache CouchDB 1.0.1
http://localhost:5984/_utils/document.html?addressbook/isom-fritsch

Řešení konfliktů

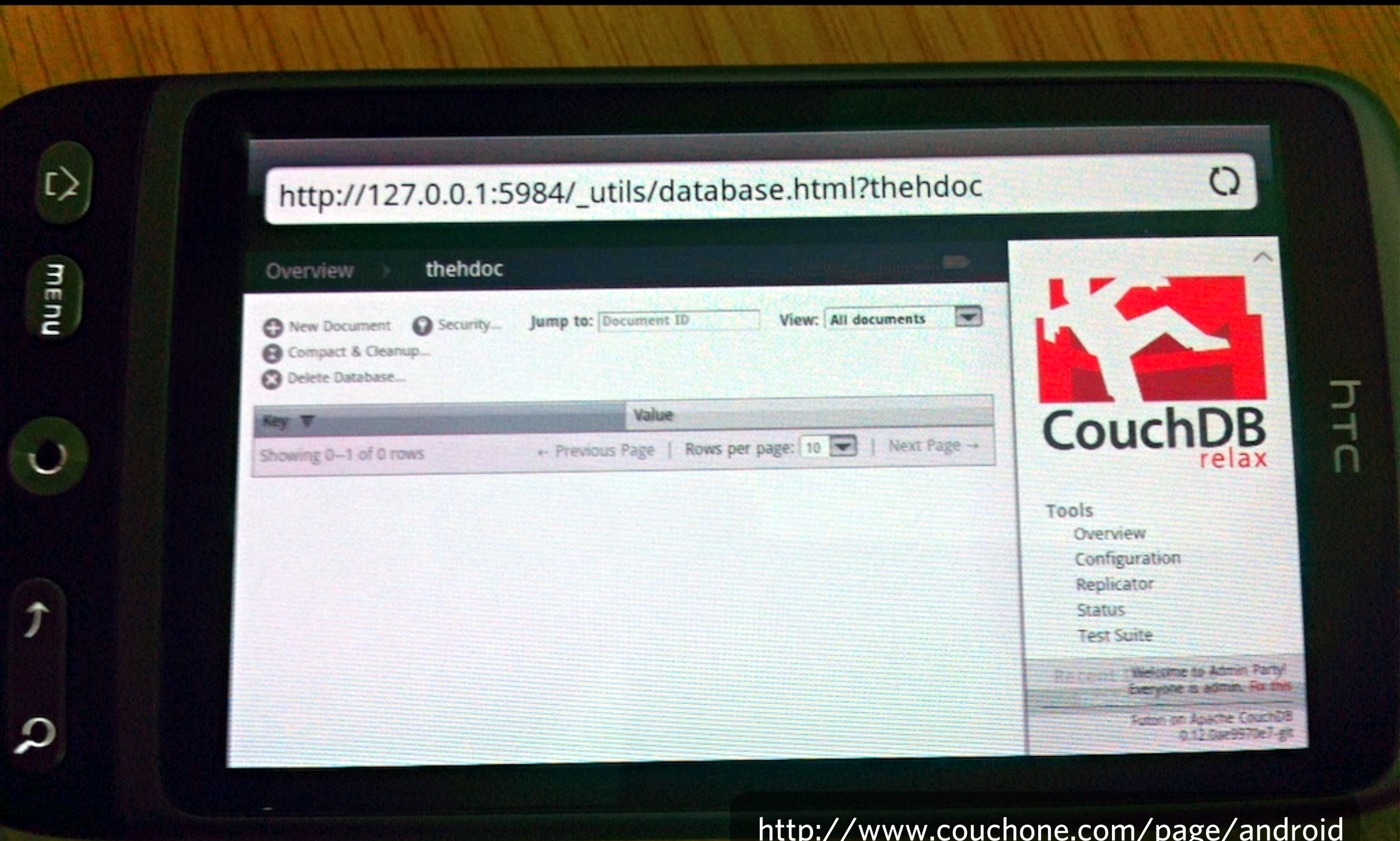


Jednoduchý multi-master cluster s Nginx

```
$ cat /opt/local/etc/couchdb/serverA.ini
...
port = 6001

$ cat /opt/local/etc/couchdb/serverB.ini
...
port = 6002

$ cat /opt/nginx/conf/nginx.conf

http {
  # ...
  upstream couchdb_cluster {
    server 127.0.0.1:6001;
    server 127.0.0.1:6002;
  }

  server {
    listen 80;
    server_name couchdb_cluster.local;

    location / {
      proxy_pass http://couchdb_cluster;
      break;
    }

  }
  # ...
}
```

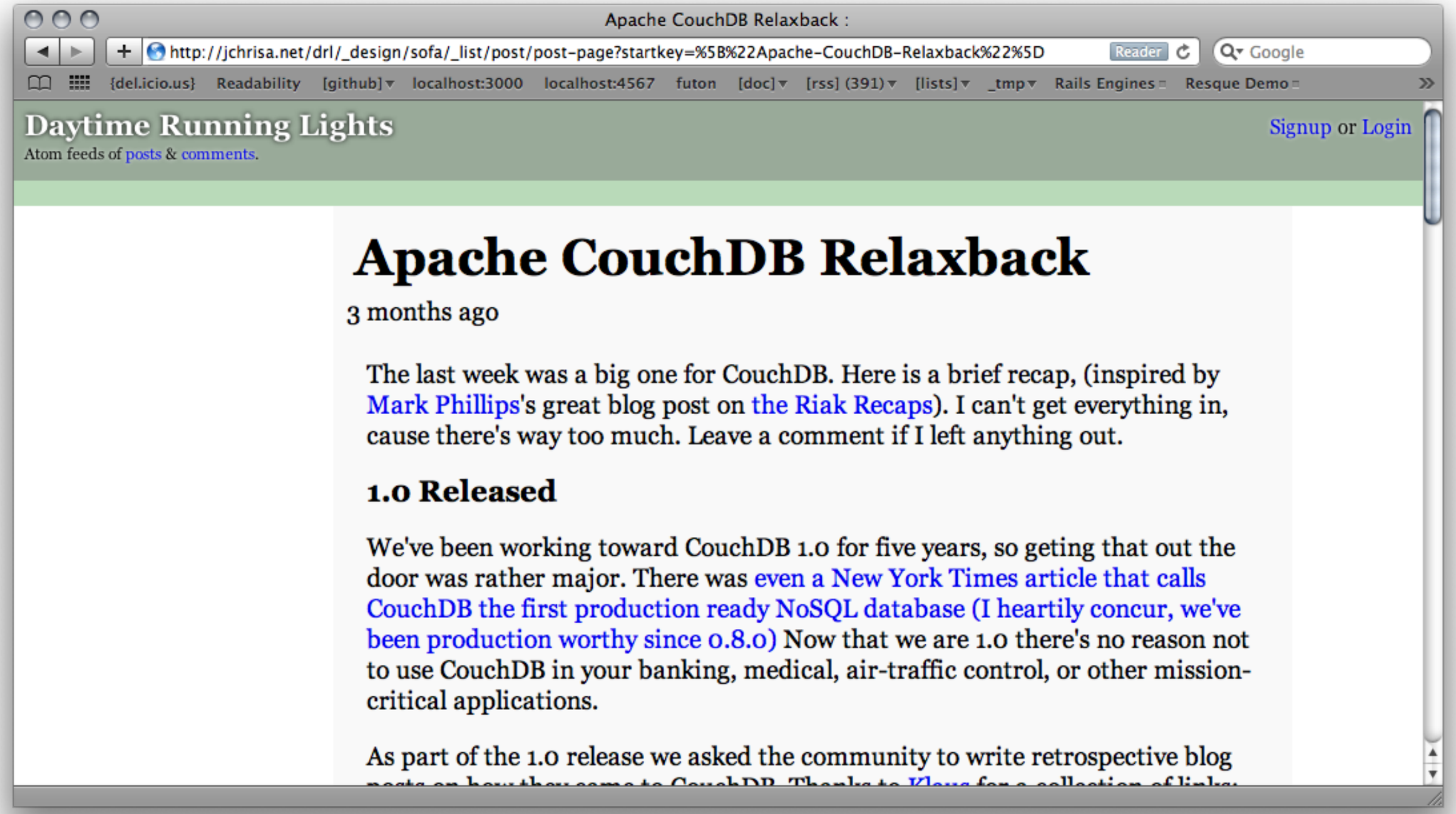
```
$ curl -X POST http://localhost:6001/_replicate \
    -d '{"source":"my_database", "target":"http://localhost:6002/my_database", "continuous":true}'

$ curl -X POST http://localhost:6002/_replicate \
    -d '{"source":"my_database", "target":"http://localhost:6001/my_database", "continuous":true}'
```

Škálování směrem **dolů**

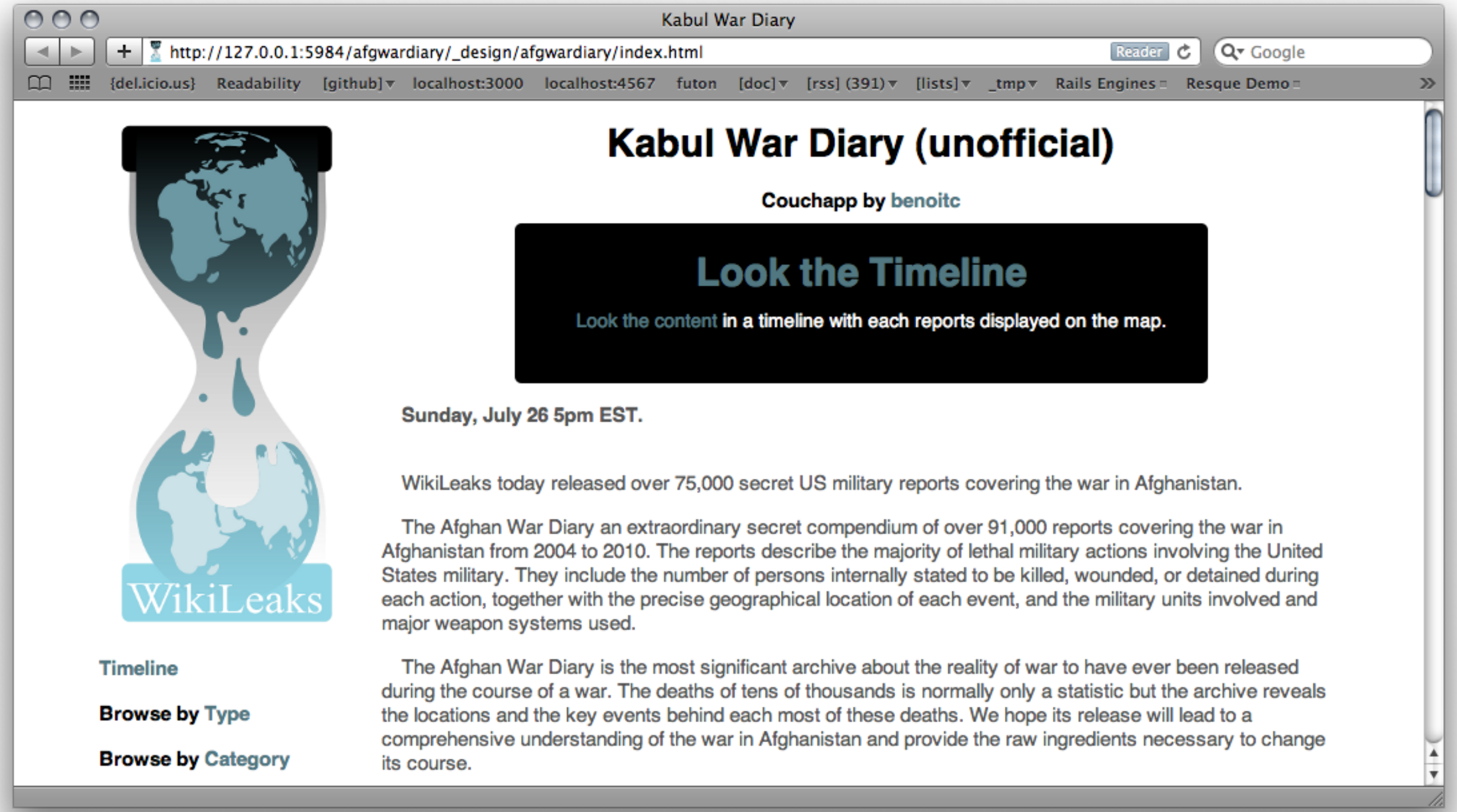


http://www.couchone.com/page/android

CouchApps

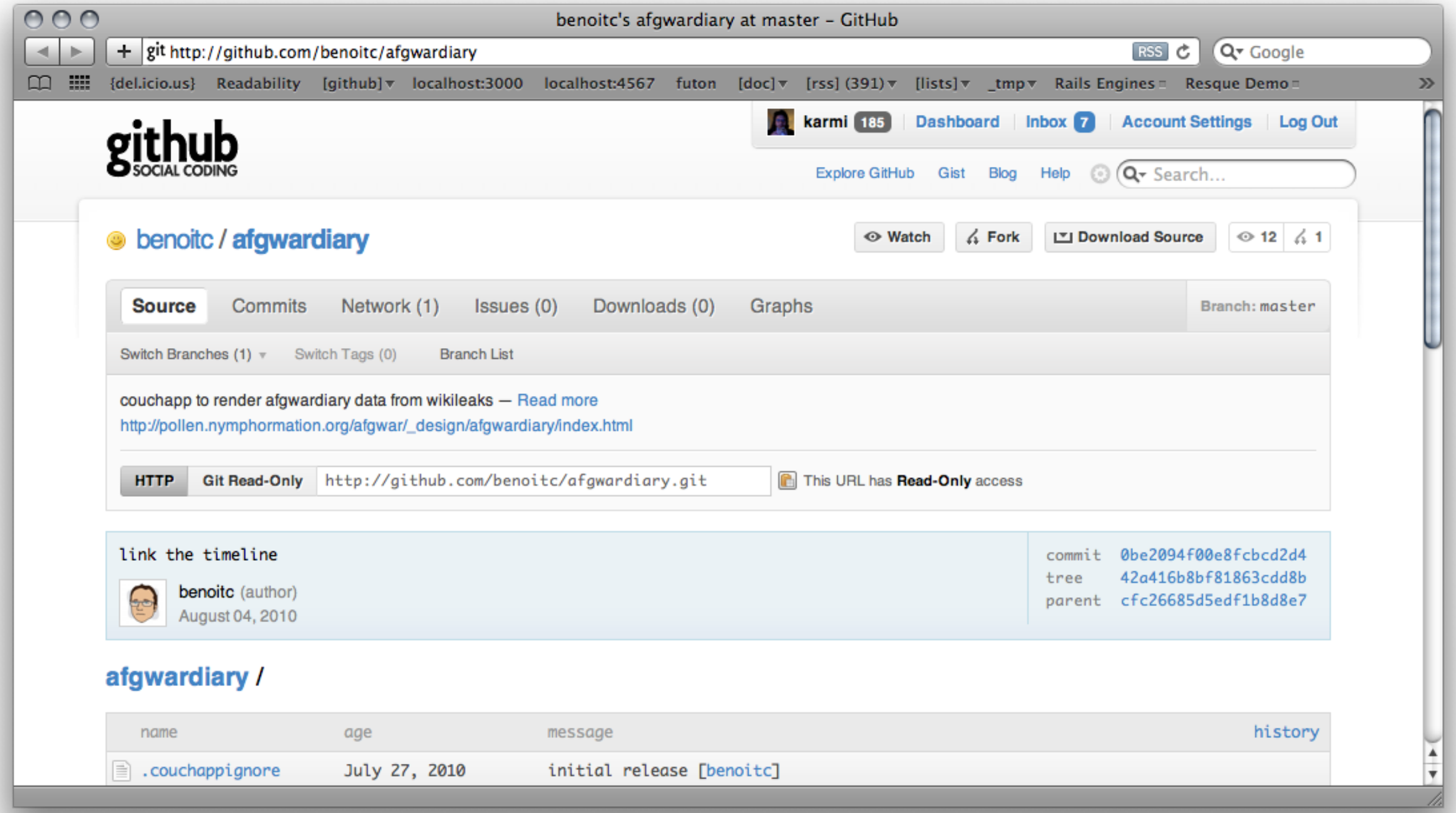

Apache CouchDB Relaxback :
http://jchrisa.net/drl/_design/sofa/_list/post/post-page?startkey=%5B%22Apache-CouchDB-Relaxback%22%5D
Reader
Google
{del.icio.us}
Readability
[github]
localhost:3000
localhost:4567
futon
[doc]
[rss] (391)
[lists]
_tmp
Rails Engines
Resque Demo
Daytime Running Lights
Atom feeds of posts & comments.
Signup or Login
Apache CouchDB Relaxback
3 months ago
The last week was a big one for CouchDB. Here is a brief recap, (inspired by Mark Phillips's great blog post on the Riak Recaps). I can't get everything in, cause there's way too much. Leave a comment if I left anything out.
1.0 Released
We've been working toward CouchDB 1.0 for five years, so geting that out the door was rather major. There was even a New York Times article that calls CouchDB the first production ready NoSQL database (I heartily concur, we've been production worthy since 0.8.0) Now that we are 1.0 there's no reason not to use CouchDB in your banking, medical, air-traffic control, or other mission-critical applications.
As part of the 1.0 release we asked the community to write retrospective blog

CouchApps



http://pollen.nymphormation.org/afgwar/_design/afgwardiary/index.html

CouchApps


benoitc's afgwardiary at master - GitHub
git http://github.com/benoitc/afgwardiary
RSS
Google
{del.icio.us} Readability [github] localhost:3000 localhost:4567 futon [doc] [rss] (391) [lists] _tmp Rails Engines Resque Demo
github
SOCIAL CODING
karmi 185 Dashboard Inbox 7 Account Settings Log Out
Explore GitHub Gist Blog Help
Search...
benoitc / afgwardiary
Watch Fork Download Source 12 1
Source Commits Network (1) Issues (0) Downloads (0) Graphs
Branch: master
Switch Branches (1) Switch Tags (0) Branch List
couchapp to render afgwardiary data from wikileaks — Read more
http://pollen.nymphormation.org/afgwar/_design/afgwardiary/index.html
HTTP Git Read-Only http://github.com/benoitc/afgwardiary.git
This URL has Read-Only access
link the timeline
benoitc (author)
August 04, 2010
commit 0be2094f00e8fcbcd2d4
tree 42a416b8bf81863cdd8b
parent cfc26685d5edf1b8d8e7
afgwardiary /
name age message history
.couchappignore July 27, 2010 initial release [benoitc]

5 Dotazování s Map/Reduce

Google

Google Research Publication: MapReduce

http://labs.google.com/papers/mapreduce.html

map reduce google

Google Research Publications — Google Labs Home

MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

Abstract

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in the paper.

Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.

Our implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable: a typical MapReduce computation processes many terabytes of data on thousands of machines. Programmers find the system easy to use: hundreds of MapReduce programs have been implemented and upwards of one thousand MapReduce jobs are executed on Google's clusters every day.

Appeared in:
OSDI'04: Sixth Symposium on Operating System Design and Implementation,
San Francisco, CA, December, 2004.

Download: PDF Version

Slides: HTML Slides

http://labs.google.com/papers/mapreduce.html

Koncept

```ruby
module Enumerable
  alias :reduce :inject unless method_defined? :reduce
end
```

```ruby
(1..3).map    { |number| number * 2 }
# => [2, 4, 6]

(1..3).reduce(0) { |sum, number| sum += number}
# => 6
```

Nejjednodušší *view*

```
function(doc) {
  if (doc.last_name && doc.first_name) {
    emit( doc.last_name + ' ' + doc.first_name, doc )
  }
}
```

Nejjednodušší *view*

INPUT

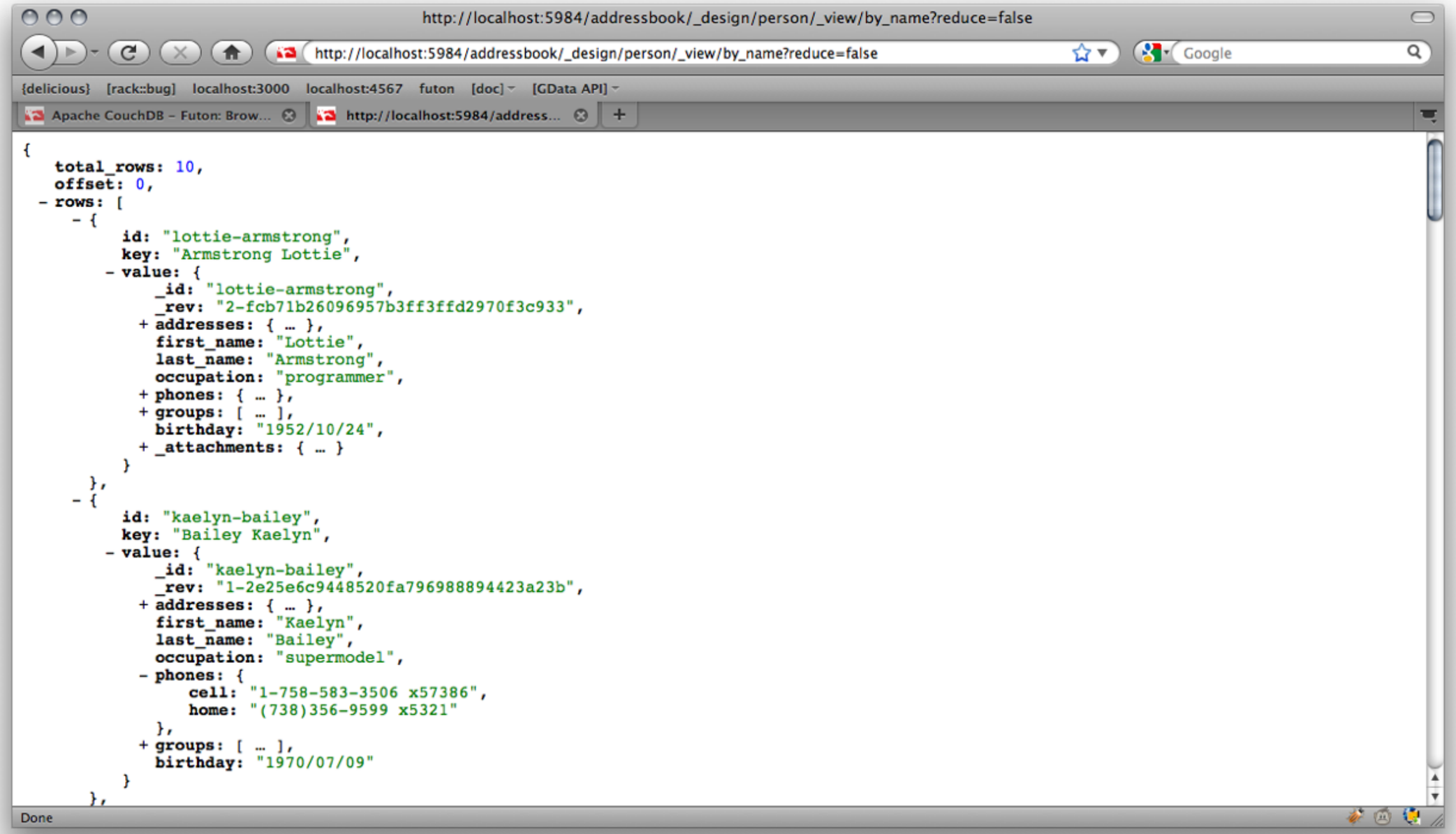
```
function(doc) {
  if (doc.last_name && doc.first_name) {
    emit( doc.last_name + ' ' + doc.first_name, doc )
  }
}
```

KEY

VALUE

OUTPUT

Výsledek: dokumenty seřazené podle příjmení



Ještě jednodušší *view*

```
function(doc) {
  emit(doc.occupation, 1);
}
```

Výsledek?

Key ▼ Grouping: exact	Value Reduce
"supermodel" ID: kaelyn-bailey	1
"supermodel" ID: jeramie-feest	1
"supermodel" ID: edd-mosciski	1
"supermodel" ID: desmond-stokes	1
"supermodel" ID: bertram-bogan	1
"supermodel" ID: andreanne-jakubowski	1
"programmer" ID: lottie-armstrong	1
"designer" ID: khalid-thiel	1
"designer" ID: jo-lubowitz	1
"designer" ID: dorothy-von	1

Showing 1-10 of 10 rows ← Previous Page | Rows per page: 50 | Next Page →

Jednoduchý *reduce*

```
function(keys, values) {
  return sum(values)
}
```

Výsledek!

Key	Grouping: exact	Value	Reduce
"supermodel"		6	
"programmer"		1	
"designer"		3	

Showing 1-3 of unknown rows ← Previous Page | Rows per page: 50 | Next Page →

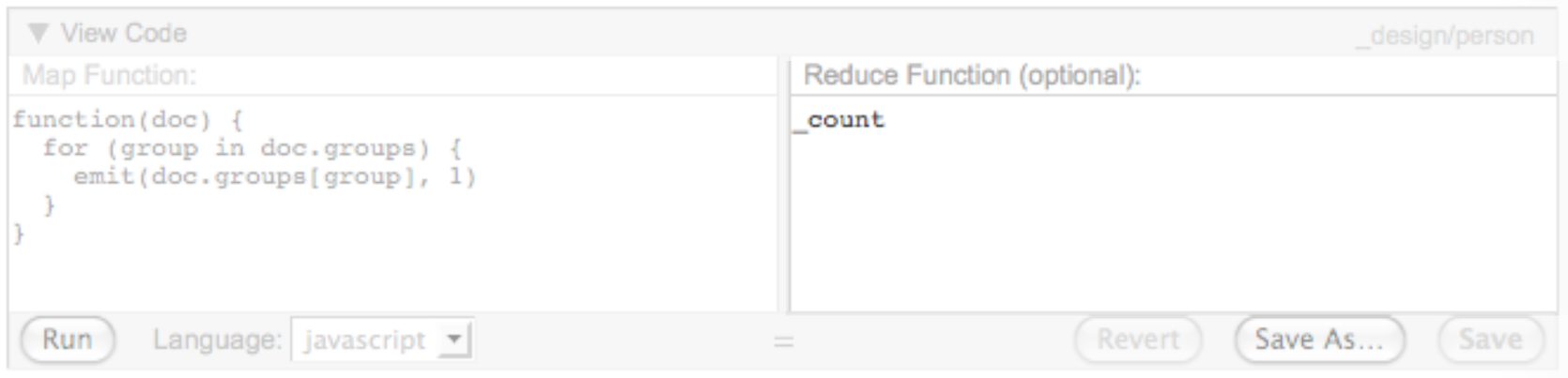
Zabudované Erlang funkce

```
$ couchdb
Apache CouchDB has started. Time to relax.
```

_count

_sum

_stats



Map/Reduce pro počítání „tagů a podobných věcí"

```
function(doc) {
  for (group in doc.groups) {
    emit(doc.groups[group], 1)
  }
}

_count
```

Výsledek *map* fáze

Key ▼ Grouping: exact	Value Reduce
"work" ID: lottie-armstrong	1
"work" ID: khalid-thiel	1
"work" ID: kaelyn-bailey	1
"work" ID: jeramie-feest	1
"work" ID: desmond-stokes	1
"friends" ID: khalid-thiel	1
"friends" ID: kaelyn-bailey	1
"friends" ID: jo-lubowitz	1
"friends" ID: dorothy-von	1
"friends" ID: andreanne-jakubowski	1
"family" ID: kaelyn-bailey	1
"family" ID: jo-lubowitz	1
"family" ID: edd-mosciski	1
"family" ID: desmond-stokes	1
"family" ID: bertram-bogan	1

Showing 1-15 of 15 rows ← Previous Page | Rows per page: 50 | Next Page →

Výsledek *reduce* fáze

Key ▽ Grouping: exact	Value ☑ Reduce
"work"	5
"friends"	5
"family"	5

Showing 1-3 of unknown rows ← Previous Page | Rows per page: 50 | Next Page →

Seskupování klíčů (*group levels*)

```
function(doc) {
  var date = new Date(doc.birthday)
  emit( [date.getFullYear(), date.getMonth()+1, date.getDate()], 1 )
}
```

COMPOSITE KEY (ARRAY)

```
_count
```

Výchozí seskupení („přesné“)

Key ▽ Grouping: exact	Value Reduce
[1990, 5, 9]	1
[1990, 5, 3]	1
[1988, 11, 20]	1
[1986, 11, 9]	1
[1983, 5, 4]	1
[1983, 2, 27]	1
[1979, 11, 21]	1
[1978, 5, 29]	1
[1975, 11, 24]	1
[1971, 5, 5]	1

Showing 1-10 of unknown rows ← Previous Page | Rows per page: 50 | Next Page →

Seskupení podle prvních dvou položek klíče

Key	Grouping: level 2	Value	Reduce
[1990, 5]		2	
[1988, 11]		1	
[1986, 11]		1	
[1983, 5]		1	
[1983, 2]		1	
[1979, 11]		1	
[1978, 5]		1	
[1975, 11]		1	
[1971, 5]		1	

Showing 1-9 of unknown rows ← Previous Page | Rows per page: 50 | Next Page →

Seskupení podle první položky klíče

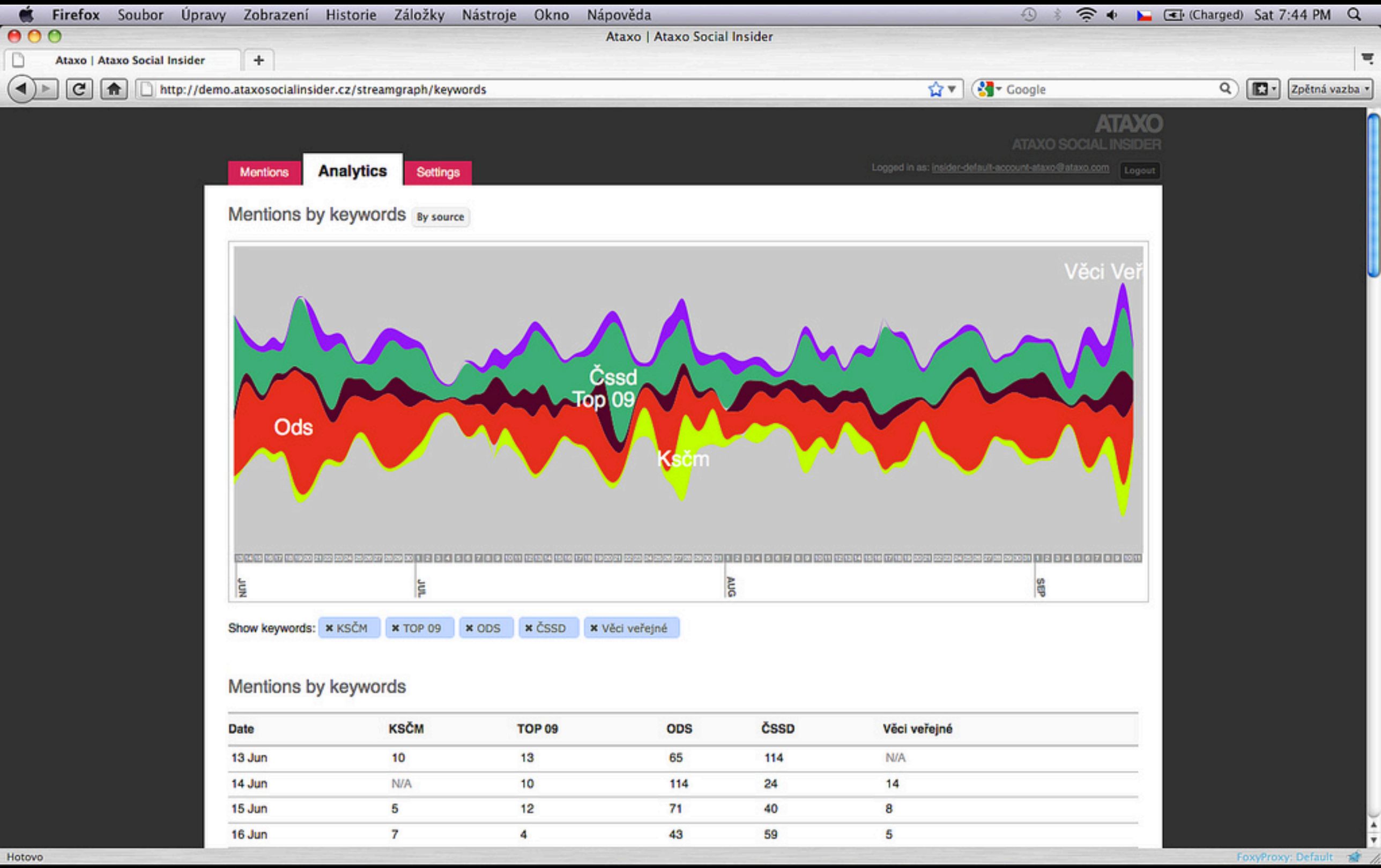
Key	Grouping: level 1	Value	Reduce
[1990]		2	
[1988]		1	
[1986]		1	
[1983]		2	
[1979]		1	
[1978]		1	
[1975]		1	
[1971]		1	

Showing 1-8 of unknown rows ← Previous Page | Rows per page: 50 | Next Page →

Parametry pro dotazování indexu

`key`
`startkey`
`startkey_docid`
`endkey`
`endkey_docid`
`limit`
`stale`
`descending`
`skip`
`group`
`group_level`
`reduce`
`include_docs`

Komplexní Map/Reduce operace



Komplexní Map/Reduce operace

```
SELECT
      COUNT(*) AS count,
        DATE_FORMAT(published_at, "%Y/%m/%d") AS date,
        keywords.value AS keyword
FROM feed_entries
  INNER JOIN feeds ON feed_entries.feed_id = feeds.id
  INNER JOIN keywords ON feeds.keyword_id = keywords.id
WHERE DATE_SUB(CURDATE(), INTERVAL 90 DAY) <= feed_entries.published_at
GROUP BY date, keyword
ORDER BY date, keyword ASC;
```

Komplexní Map/Reduce operace

Jenže. Já nechci „řádky“. Já chci data zhruba v tomto formátu:

```
Streamgraph.load_data({
  max : 170,
  keywords : ['ruby', 'python', 'erlang', 'javascript', 'haskell'],

  values   : [
    { date: '2010/01/01', ruby: 50,  python: 20, erlang: 5,  javascript: 30, haskell: 50 },
    { date: '2010/02/01', ruby: 20,  python: 20, erlang: 2,  javascript: 40, haskell: 43 },
    { date: '2010/03/01', ruby: 70,  python: 20, erlang: 10, javascript: 80, haskell: 15 },
    { date: '2010/04/01', ruby: 20,  python: 40, erlang: 8,  javascript: 30, haskell: 12 },
    { date: '2010/05/01', ruby: 150, python: 30, erlang: 12, javascript: 40, haskell: 18 },
    { date: '2010/06/01', ruby: 30,  python: 10, erlang: 14, javascript: 17, haskell: 14 }
  ]

});
```

Fáze *map*

```
function(doc) {
  // Fix potentially junk date formats in docs
  var fix_date = function(junk) {
      var formatted = junk.toString().replace(/-/g,"/").replace("T"," ").substring(0,19);
      return new Date(formatted);
    };
  // Format integers to have at least two digits.
  var f = function(n) { return n < 10 ? '0' + n : n; }
  // This is a format that collates in order and tends to work with
  // JavaScript's new Date(string) date parsing capabilities, unlike rfc3339.
  Date.prototype.toJSON = function() {
      return this.getUTCFullYear()   + '/' +
             f(this.getUTCMonth() + 1) + '/' +
             f(this.getUTCDate())      + ' ' +
             f(this.getUTCHours())     + ':' +
             f(this.getUTCMinutes())   + ':' +
             f(this.getUTCSeconds())   + ' +0000';
  };
  if (doc['couchrest-type'] == 'Mention') {
    for ( keyword in doc.keywords ) {
      var key = fix_date(doc.published_at).toJSON().substring(0,10);
      var value = {};
      value[ doc.keywords[keyword] ] = 1;
      emit( key, value);
    }
  }
}
```

Fáze *reduce*

```
function(keys, values, rereduce) {
  if (rereduce) {
    var result = {}
    for ( item in values ) {
      for ( prop in values[item] ) {
        if ( result[prop] ) { result[prop] += values[item][prop]  }
        else { result[prop] = values[item][prop]  }
      }
    }
    return result;
  }
  else {
    // Prepare the data for the re-reduce
    var result = {}
    for ( value in values ) {
      var item = values[value];
      for ( prop in item ) {
        if ( result[prop] ) { result[prop] += item[prop] }
        else { result[prop] = item[prop] }
      }
    }
    return result;
  }
}
```

Výsledek

```
$ curl http://localhost:5984/customer_database/_design/Mention/_view/by_date_and_keyword?group=true
```


```
{
    "rows": [
        {
            "key": "2010/09/22",
            "value": { "ruby": 8, "python": 19 }
        },
        {
            "key": "2010/09/23",
            "value": { "ruby": 24, "python": 12 }
        },
        {
            "key": "2010/09/24",
            "value": { "ruby": 7, "python": 8 }
        }
    ]
}
```

I♥JS. Nebo... ne?


wtfjs
http://wtfjs.com/2010/02/13/string-is-not-string
RSS
Google
{del.icio.us} Readability [github] localhost:3000 localhost:4567 futon [doc] [rss] (418) [lists] _tmp
wtfjs
about
rss
archive
Fork me on GitHub
string is not string
"string" instanceof String; // false.
// 'course it isn't not a string, it may look like a string
// but actually it's masquerading as a banana.
When is a string, not a string? When it's a duck - @rem
wtfjs is free software. get the source on github.

Komplexní dotazy

No… a co když potřebuji něco jako:

Vypiš všechny programátory z Plzně.

Smůla?

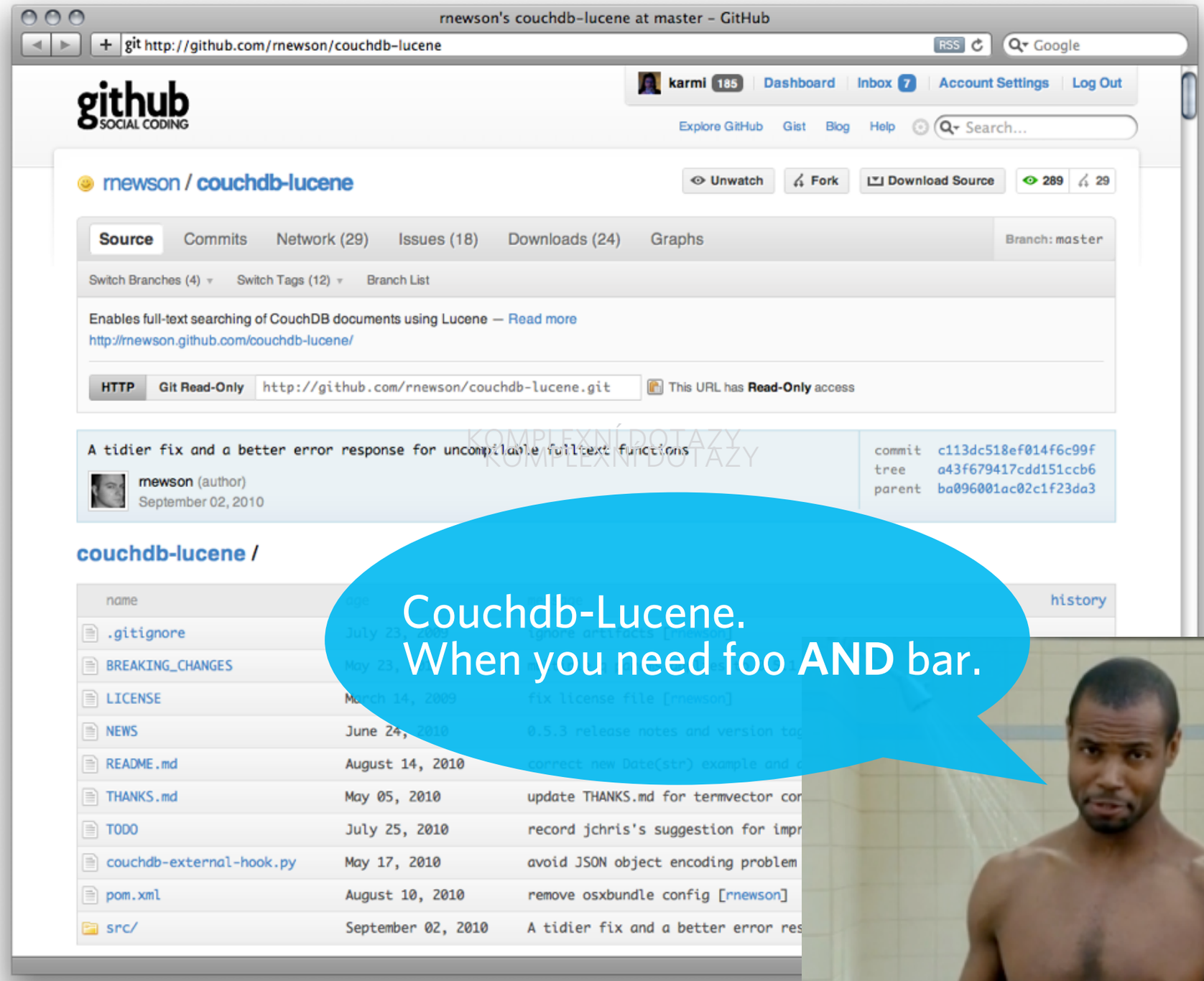
CouchDB-Lucene

Robert Newson

Tenhle chlapík to ví.

Vypiš všechny programátory z Plzně.

foo AND bar



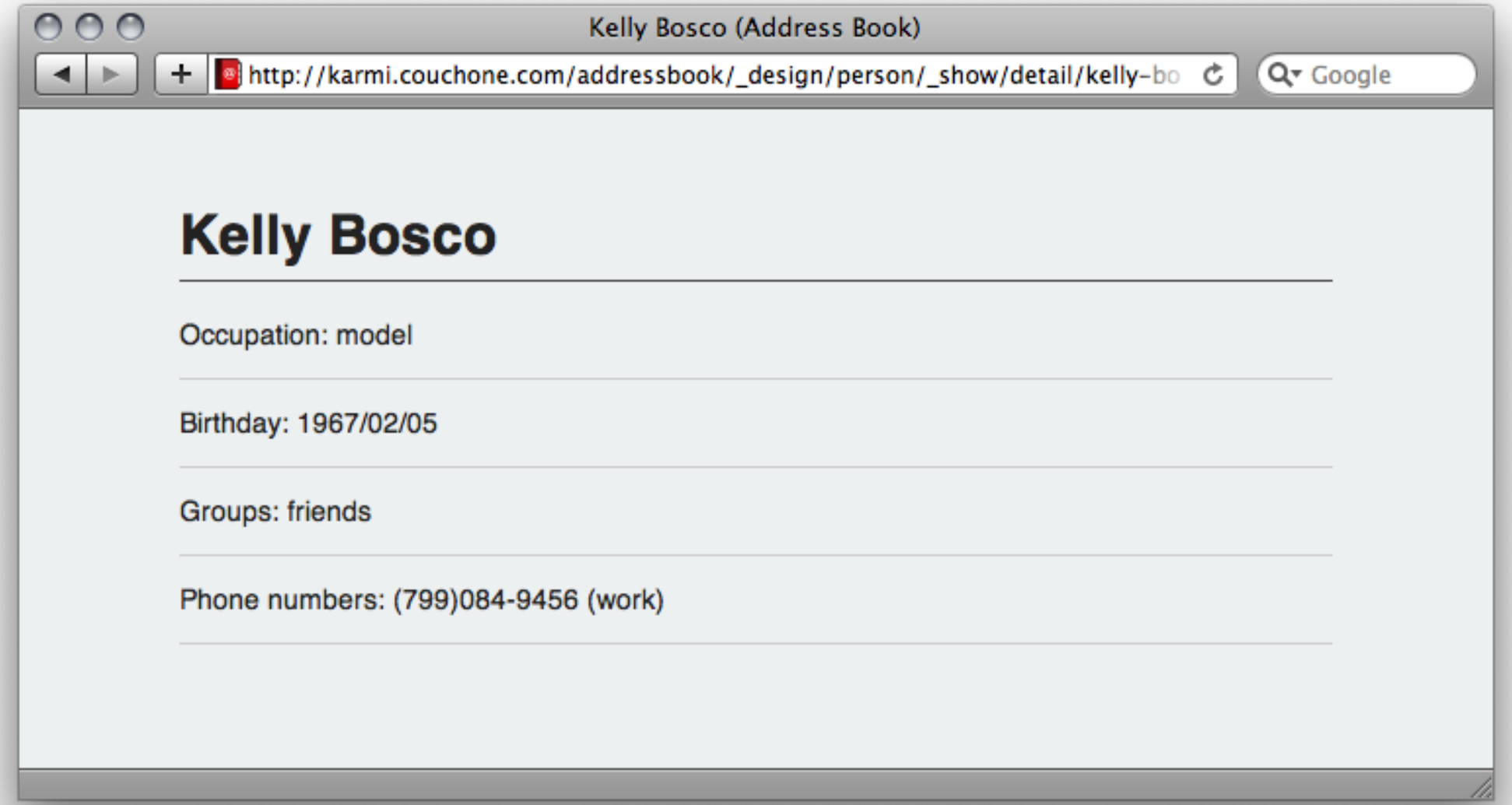
http://github.com/rnewson/couchdb-lucene

Definice indexu

```
function(doc) {

  var result = new Document();

  if (doc.occupation)     {
    result.add(doc.occupation,                    {"field":"occupation"})
  }

  if (doc.addresses)      {
    for (address in doc.addresses) {
      result.add(doc.addresses[address].city,     {"field":"city"})
    }
  }

  return result;

}
```

http://localhost:5984/addressbook/_fti/_design/person/search?q=occupation:programmer AND city:Plzeň

Ukázková aplikace



SOURCE CODE: http://github.com/karmi/couchdb-showcase

Resources

- http://guide.couchdb.org
- https://nosqleast.com/2009/#speaker/miller
- http://www.couchone.com/case-studies
- http://www.couchone.com/migrating-to-couchdb
- http://wiki.apache.org/couchdb/
- http://blog.couchone.com/
- http://stackoverflow.com/tags/couchdb/

Otázky, prosím!