CouchDB

relax

Niklas Gustavsson

niklas.gustavsson@callistaenterprise.se
www.callistaenterprise.se

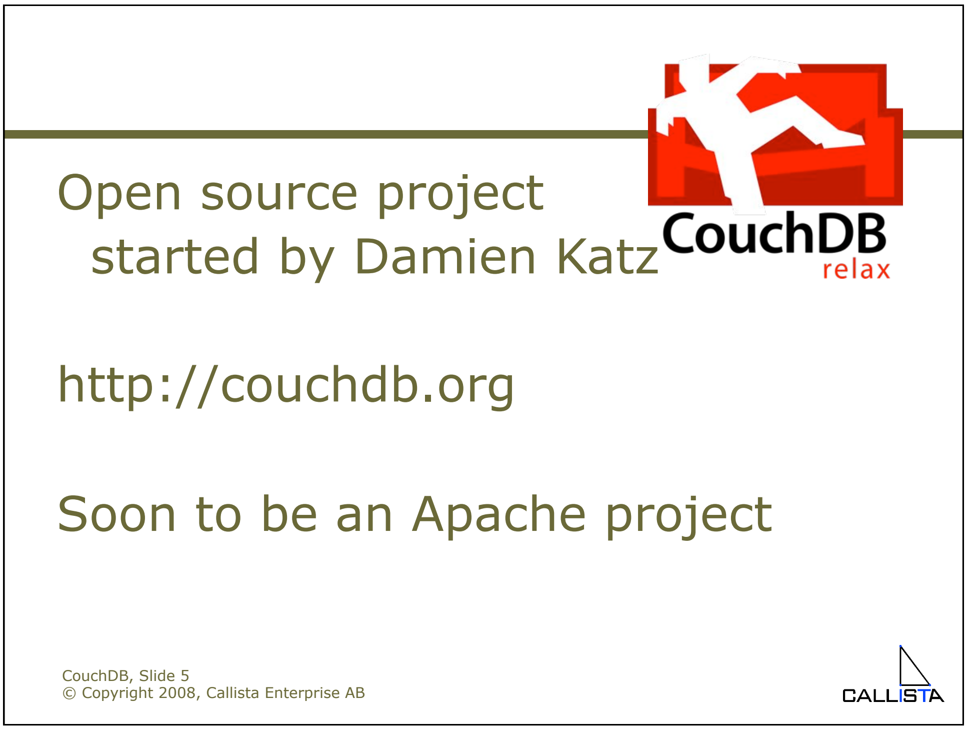
CALLISTA

Be warned

Won't replace your relational database

You (probably) won't be using it any time soon

Forget about relational databases for a while



Open source project
started by Damien Katz

http://couchdb.org

Soon to be an Apache project

How does it work?

Document oriented

Documents are JSON

```
{firstName:”Niklas”,
  lastName: “Gustavsson”}
```

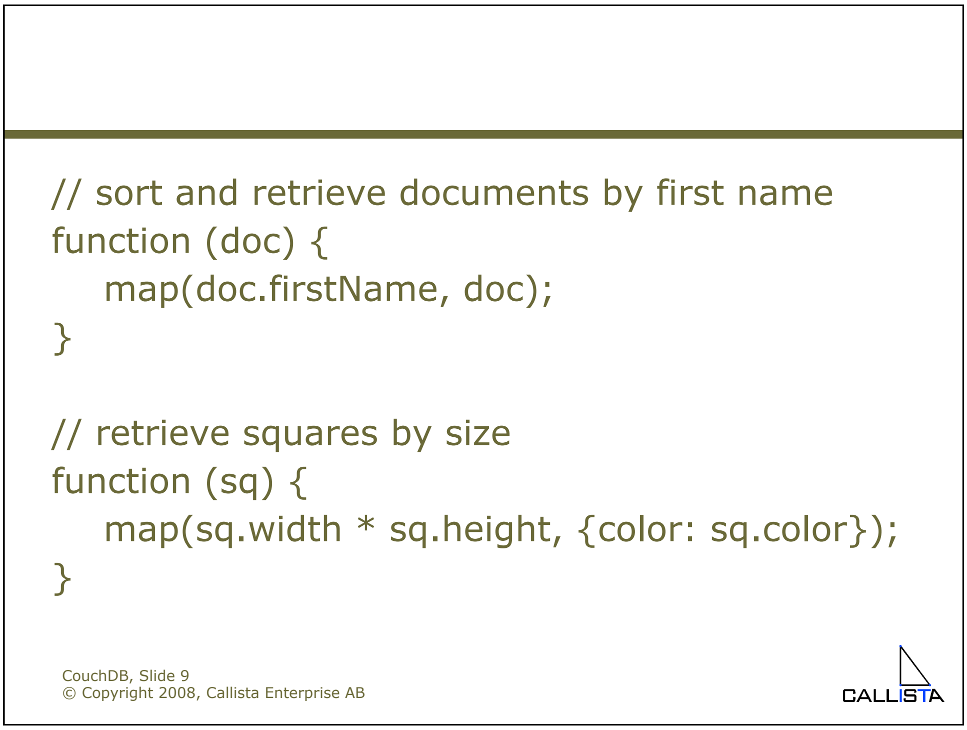
Schema less

Views

Pre-computed, indexed table

Incrementally updated

Written in JavaScript



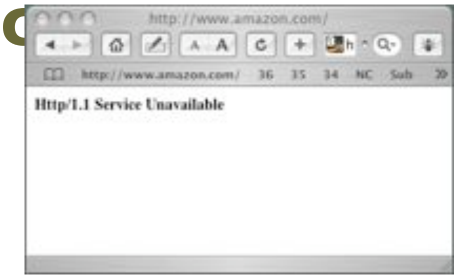
```
// sort and retrieve documents by first name
function (doc) {
    map(doc.firstName, doc);
}

// retrieve squares by size
function (sq) {
    map(sq.width * sq.height, {color: sq.color});
}
```

So why should I care?

1 2 3

First reason

Sometimes availability trumps consistenc



Http/1.1 Service Unavailable

CAP theorem - pick two:

Consistency
Availability
Partitioning

Eventual Consistency

http://www.allthingsdistributed.com/2007/12/eventually_consistent.html

NODES – the number of nodes that store a replica

WRITES – the number nodes that confirm a commit

READS – the number of nodes that are contacted at a read operation

WRITES+READS > NODES : strong consistency

http://www.allthingsdistributed.com/2007/12/eventually_consistent.html

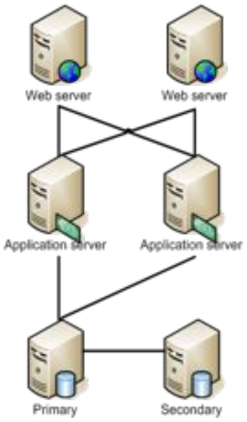
Synchronous DB replication

NODES = 2

WRITES = 2

READS = 1

2 + 1 > 2 → Strong consistency



http://www.allthingsdistributed.com/2007/12/eventually_consistent.html

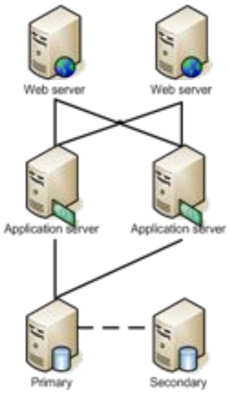
Asynchronous DB replication

NODES = 2

WRITES = 1

READS = 1

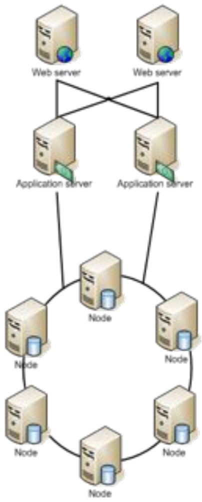
1 + 1 > 2 → Eventual consistency



http://www.allthingsdistributed.com/2007/12/eventually_consistent.html

What if NODES is 10 or 100 or 1000?

Every increase in WRITES means less chance for a write to succeed



Read more

http://www.allthingsdistributed.com/2007/12/eventually_consistent.html

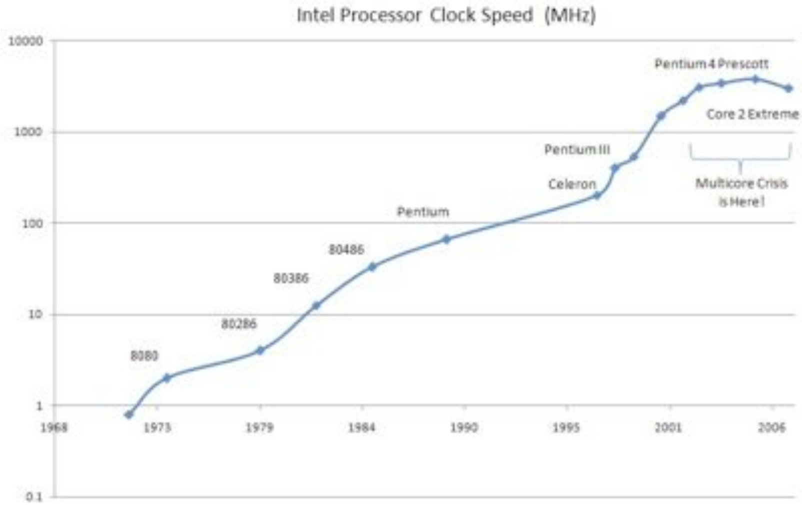
http://aws.amazon.com/simpledb

http://lucene.apache.org/hadoop/

1 2 3

Second reason:

Moore has changed tactic



CPUs are getting slower but more plenty

The Free Lunch is over - http://www.gotw.ca/publications/concurrency-ddj.htm

We need to start parallelizing our tasks

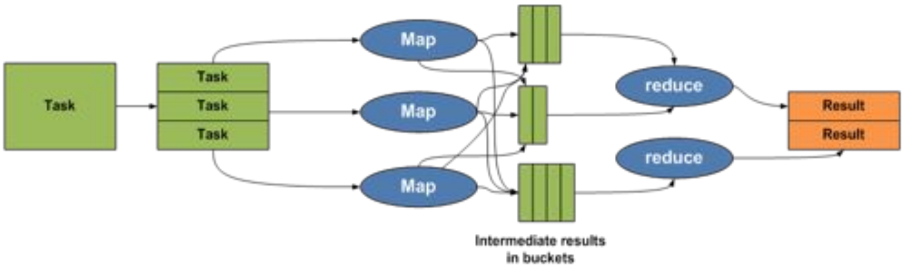
Concurrency in Java is really, really hard

Share nothing

- Erlang processes and messaging
- Map/reduce

map/reduce

Used for computing views in CouchDB



Read more

http://www.gotw.ca/publications/concurrency-ddj.htm

http://erlang.org/

http://www.scala-lang.org/

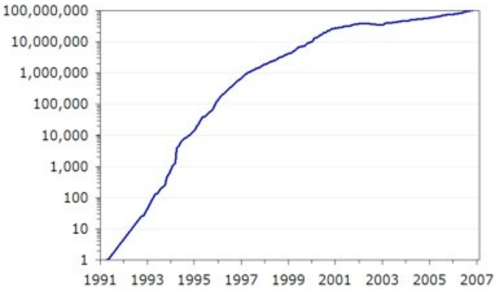
http://labs.google.com/papers/mapreduce.html

CALLISTA

1 2 3

Third reason

Web friendly



CALLISTA

REST - “HTTP used right”

REST based API

- Everything is a resource
- Every resource has a URL
- Every resource has the same uniform interface
- Links guides through states
- Stateless

REST freebies

API easily usable from any platform (AJAX , Java, Ruby, COBOL, Powerpoint...)

Works with existing infrastructure

Caches, proxies, firewalls…

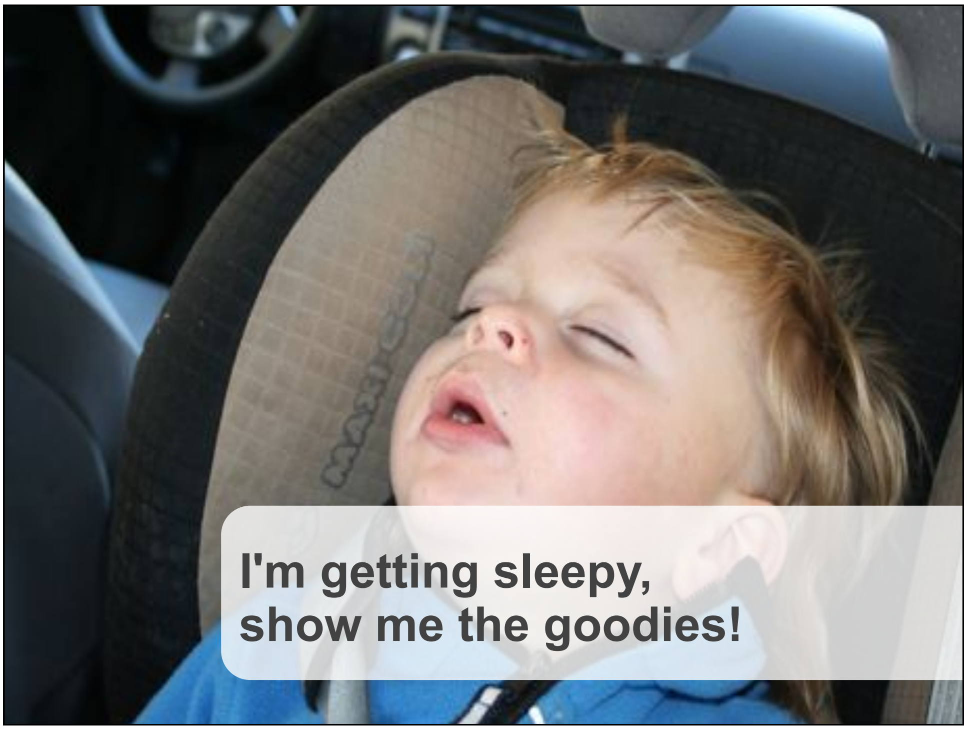
Optimistic locking

Read more

http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

http://www.infoq.com/articles/rest-introduction

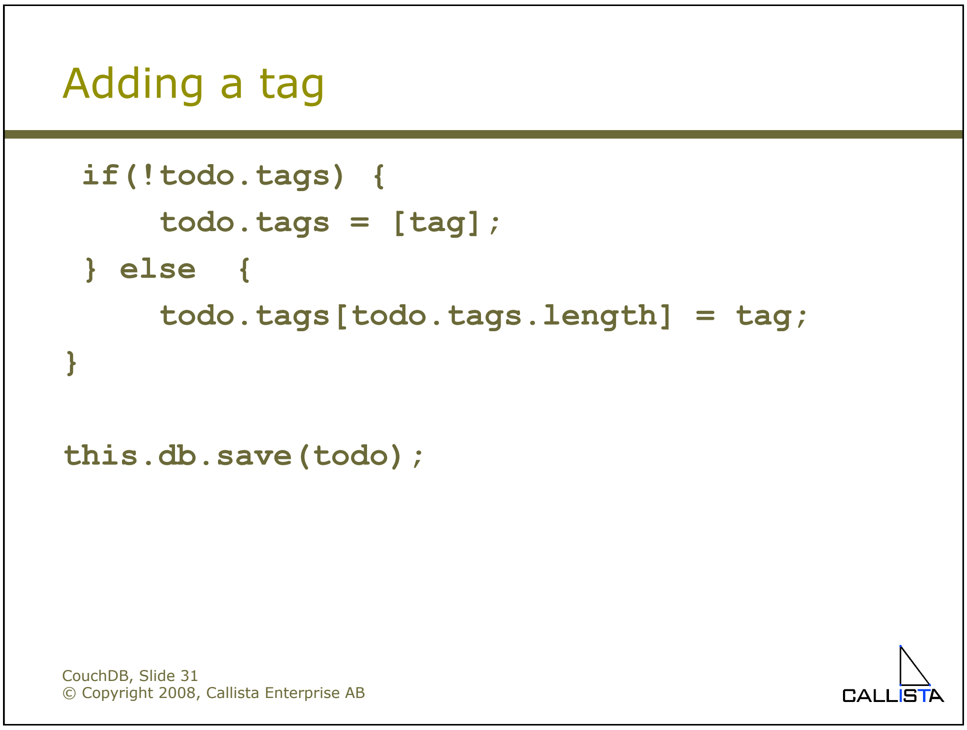
http://www.burtongroup.com/Guest/Aps/RestWorkshop.aspx



I'm getting sleepy,
show me the goodies!

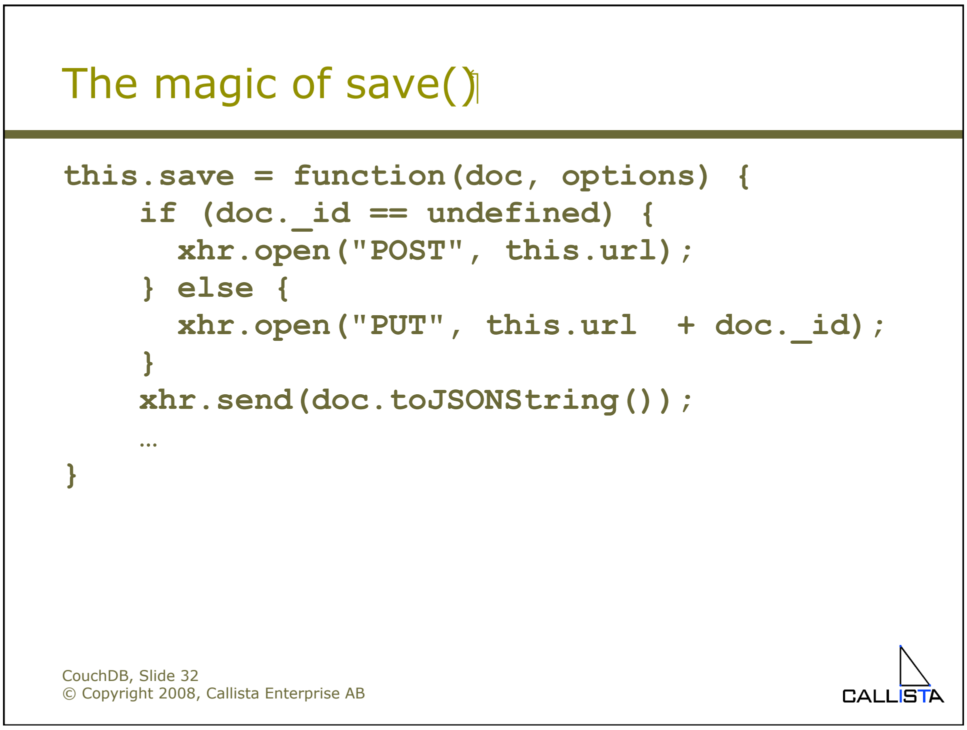
Adding a to do

```
var todoText = $("todoinput").value;
this.db.save({text: todoText });
```



Adding a tag

```
 if(!todo.tags) {
     todo.tags = [tag];
 } else  {
     todo.tags[todo.tags.length] = tag;
}

this.db.save(todo);
```



The magic of save()

```
this.save = function(doc, options) {
    if (doc._id == undefined) {
      xhr.open("POST", this.url);
    } else {
      xhr.open("PUT", this.url  + doc._id);
    }
    xhr.send(doc.toJSONString());
    …
}
```

Questions?

?

Attributions

WHY? - http://www.flickr.com/photos/teflon/128827389/

Deep mud - http://www.flickr.com/photos/hubmedia/133598031/

Banana - http://flickr.com/photos/tim_ellis/154225908/

Forget me not - http://flickr.com/photos/doblonaut/456339900/