Next Generation Data Storage with CouchDb

Jan Lehnardt <jan@php.net>

About Me

- Freelancing Developer with strong focus on the Web for ~6 Years (see notes, Jan)
- Studies in Computer Science and Linguistics (not finished :)
- Co-founder of FREISATZ
- Love Rock music and kittens

About You?

- What do you work on?
- What are you interested in?
- Why are you here?

Target Audience

- (Let’s see how well this matches the responses to the last slide)

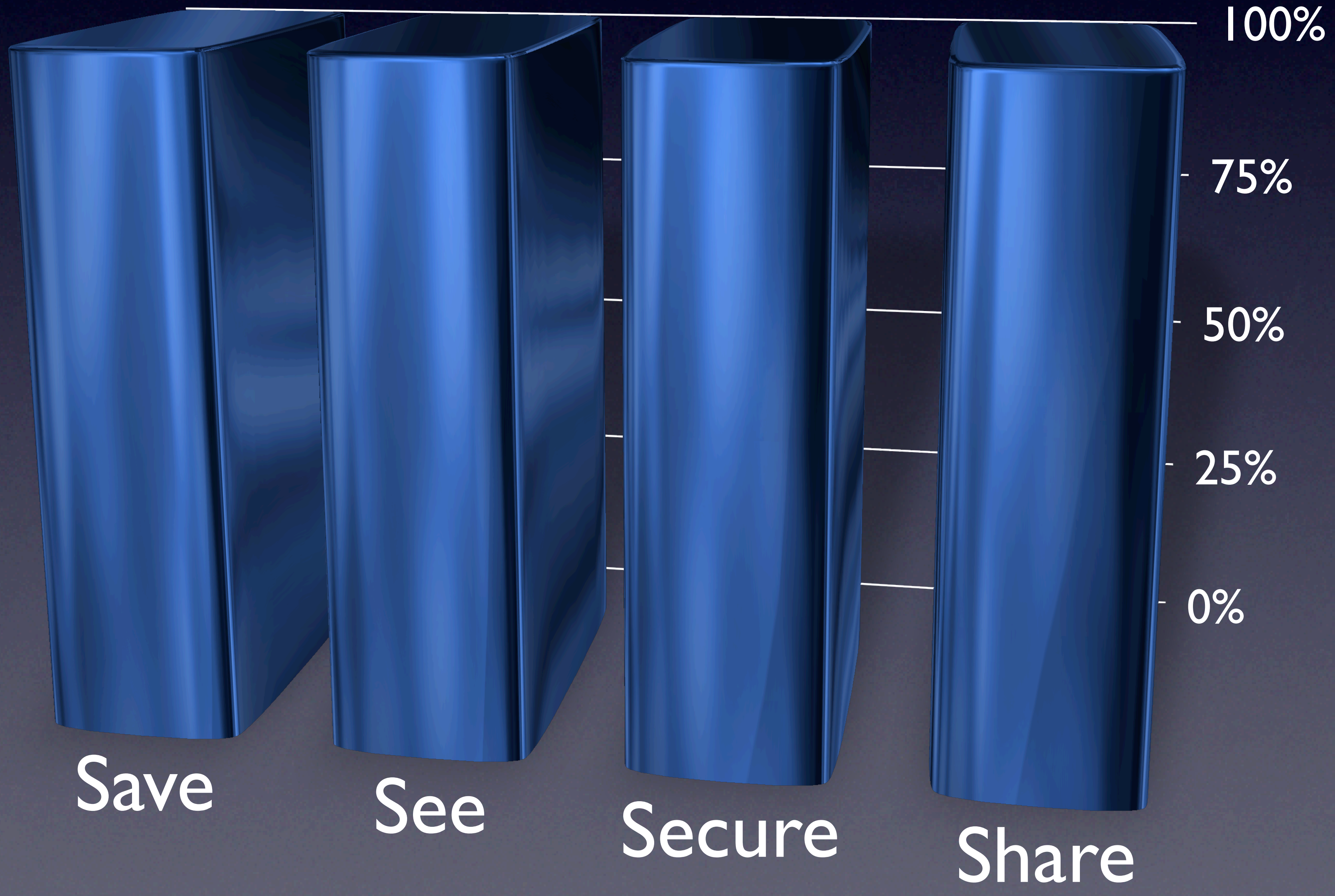
- Web-Developers, interested in new technology
  - Others work too

Forget SQL

- Just for a moment, imagine you know nothing about relational data, normalization and other fancy terminology

- You only want to store data

The Four Pillars of Data Management



Four Pillars (1): Save

- Save data securely, permanent etc...
  - i.e. ACID compliant
- Save data efficient

Four Pillars (2) See

- Make data available for easy retrieval
- Integrate simple reporting methods
- Provide (fulltext) search

Four Pillars (3) Secure

- Enable strong compartmentalization of data
- Assign users, groups and roles to data
- Allow connections over SSL

Four Pillars (4) Share

- To make it all complete:
    - Do all that in a distributed way
    - On- and Offline(!)

No Relations

- Enter CouchDb
  - New implementation, no legacy
  - Designed to…
    - …implement the four Pillars
    - …leverage recent paradigm shifts

CouchDb Features

Robust Data Storage	Replication
REST API	User Authentication
Fabric Views	Built on Erlang/OTP
Fulltext Search	Your Feature here, it’s Open Source!

Data Storage

- State-of-the-art storage model
  - MVCC with optimistic locking
    - Scales magnitudes better than pessimistic locking
- Append-only writes

Documents

- Individual storage elements
  - Can have an arbitrary structure…
  - …but don’t have to!
  - Attachments
- Have IDs & revisions

REST API

- Easy retrieval using a method proven to be scalable: HTTP
- Exchange in industry-standard formats (XML/JSON)
- Simple and intuitive interface
    - Demo

Fabric / Rum(mage)

- Simple language to define Views
- Done for XML model
- Work-in-progress for JSON
- Planned: Structural pattern-matching for documents (hot stuff!)

Fulltext Search

- Leverages Lucene
  - Reference implementation included
- Simple API to plug in other search technology
  - Show them

Security and Authentication

- Not yet implemented
- Will have
  - Standard user-authentication for applications
  - Document ownership

Replication

- Bi-directional replication (or synchronization) between any number of instances
  - Online, and:
  - Offline
    - Including consistent conflict resolution for disconnected systems. Wow!

Erlang Core

- Multi-everything
- Lightweight threading
- Awesome performance
- Die-on-error design
- Awkward syntax, but hey :-)

CouchDb for Database Experts

Now, remember again SQL and all you know about traditional database systems

Databases

Documents

- Individual storage elements
  - Can have an arbitrary structure…
  - …but don’t have to!
  - Attachments
- Have IDs & revisions

Example Document

```
<?xml version="1.0"?>
  <doc id="2B3B17B0C93D2461DFEF714655E67F70" rev="4134847"
  xmlns="http://couchdb.org/doc">

    <field name="BugId">
      <text>098F6BCD4621D373CADE4E832627B4F6</text>
    </field>
    <field name="Counter">
      <text>1</text>
    </field>
    <field name="Name">
      <text>test</text>
    </field>
    <field name="Type">
      <text>Project</text>
    </field>

  </doc>
```

Example Document

```
{
  id:”098F6BCD4621D373CADE4E832627B4F6”,
  rev: 4134847,
  fields: [
    {“BugId”: “098F6BCD4621D373CADE4E832627B4F6”},
    {“Counter”:1},
    {“Name”:”test”},
    {“Type”:”Project}
  ]
}
```

Views

- Similar to Views
  - Compiled pre-defined subset of stored data

Code

```
<?php
include 'CouchDb/Couch.php';
$couch = new Couch($host, $db);
$couch->running() or die(‘CouchDb offline’);

$doc = new CouchDocument;
$doc->set(‘Name’, ‘Derick’);
$doc->age = 27;
$couch->save($doc);
?>
```

Code

```
...
$table = <<<EOD
  SELECT *
  COLUMN Name
  COLUMN Age
EOD;
$couch->table_add('dudes', $table);

foreach(list($name, $age) = $couch->get_all('dudes')) {
  echo “$name ist $age alt\n”;
}
...
```

Cool things to do with CouchDb

- Distributed Anything-Application
- The Wiki
- Version Control System
- FUSE plugin
- MySQL Storage Engine

About Damien

- He did it all!
- Ex-Iris/Lotus/IBM
  - Helped creating Lotus Notes Database
- Took two years off to build CouchDb
- Now at MySQL

Literature

http://couchdb.com/

http://couchdbwiki.com/

http://damienkatz.net/

http://jan.prima.de/plok/

http://google.com?q=Thinking%20in%20Erlang

The Book

Cheers!

Cheers!

You’ve been great!