



CouchDB for (absolute) beginners

# CouchDB for (absolute) beginners

NoSQL Dundee Conference 2010

Simon Wells

School of Computing, University of Dundee

[swells@computing.dundee.ac.uk](mailto:swells@computing.dundee.ac.uk)

<http://www.simonwells.org>

# About Me

- Simon Wells
- [swells@computing.dundee.ac.uk](mailto:swells@computing.dundee.ac.uk)
- <http://www.simonwells.org>
- Research Fellow @ School of Computing
- Research Areas: {Artificial Intelligence, MultiAgent Systems, Defeasible Reasoning}
- Teaching: {Argumentation & Critical Thinking, Core Computer Science Topics, MultiAgent Systems, Grid & Cloud Computing}



# Why Me?

- Web-scale argumentation
- Building a system on (quick)sand
- Not sure of extent/structure of data
- Mistake!

# Overview

- CouchDB-in-a-slide
- History
- Overview
- Document Orientation
- RESTful API
- Using CouchDB with curl
- Using CouchDB with Futon
- Views
- Some background
- Some resources

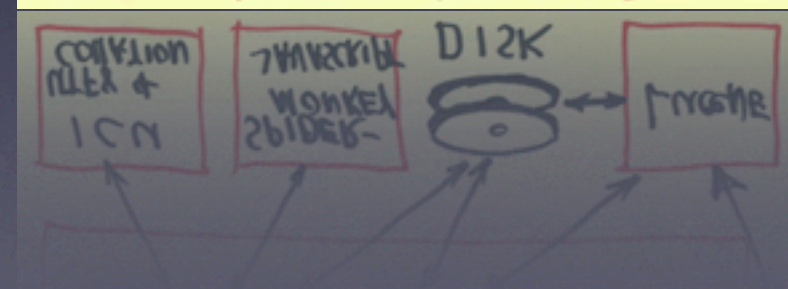
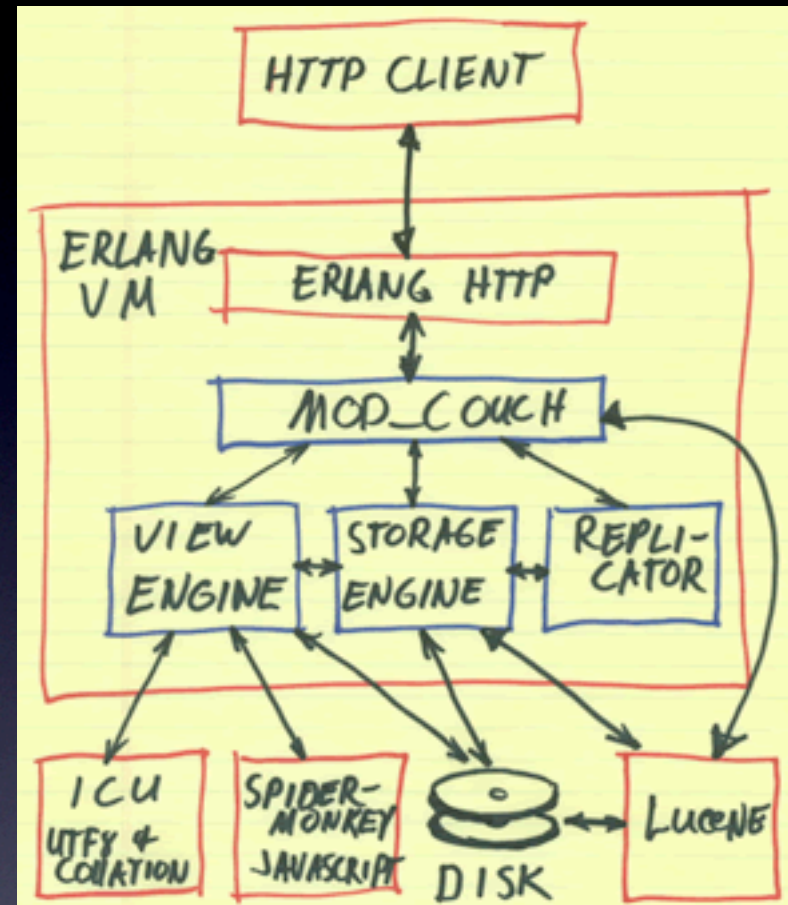
# CouchDB in-a-slide

- COUCH: Cluster Of Unreliable Commodity Hardware
- Document-oriented database
- Queried & indexed *a la* MapReduce using Javascript
- RESTful JSON API
  - can access from anything that supports HTTP requests
    - e.g. curl for core API
  - Futon built-in web-based admin console
  - third-party client libraries
- Incremental replication
  - with bi-directional conflict-detection & resolution

# History of CouchDB

- 2005, Damien Katz, formerly of IBM publishes a blog post about his new database engine that would be called CouchDB:
  - “storage system for a large-scale object database”
- Plan:
  - To become **the** Internet database
  - Designed ground-up to serve web-apps
- Written in C++ then moved to Erlang/OTP due to latter’s focus on fault tolerance (9x9s uptime for a switching system at Ericsson)
- 2008: (Feb) Apache Incubator project (Nov) Top-level Apache project
- Maintained by Apache Software Foundation
- Supported by IBM
- Katz still lead developer

# System Overview



# Document Oriented

- “any discrete representation of meaning”
- readable, writeable, addressable
- RESTful:
  - client-server | stateless | cacheable | layered | uniform interface

# CouchDB Documents

- JSON objects
  - Name/value pairs
  - Arbitrary nested structure
- JSON
  - *JavaScript Object Notation*
  - Lightweight data interchange format based on Javascript syntax
- Can contain file attachments
  - unlimited (practically) in number & size
  - one CouchDB doc can contain an entire web page: HTML, metadata, images, video, audio, flash, &c.

# Key/Value Store

Key	“user-simon-office-tel-num”
Value	“01382-386-526”

# Example: Business Cards

name	Simon Wells
business	University of Dundee
phone (office)	01382-386-526
email	<u><a href="mailto:swells@computing.dundee.ac.uk">swells@computing.dundee.ac.uk</a></u>
IM	<u><a href="mailto:sw3lls@yahoo.co.uk">sw3lls@yahoo.co.uk</a></u>
Skype	si.wells
address (office)	University of Dundee, Dundee, DDI 4HN

# A CouchDB Document in JSON

```
{  
  "_id": "4AD29BBB737bc5264243E23816BE3765",  
  "_rev": "64BB2EB8",  
  "type": "",  
  "firstname": "Simon",  
  "lastname": "Wells",  
  "company": "University of Dundee",  
  "email": "swells@computing.dundee.ac.uk",  
  "IM": "sw3lls@yahoo.co.uk",  
  "Skype": "si.wells",  
  "OfficeAddress": "Queen Mother Building",  
  ...  
}
```

# RESTful HTTP API

- Complete & Extensive API available via HTTP:
  - GET/PUT/POST/DELETE
  - Can work with couchdb entirely using curl
  - Can use a web-app (Futon)
  - Can use client libraries from favourite language (or raw HTTP calls)

# CouchDB with curl

- Assuming you know the IP address for your couchDB install:
  - `$ curl http://127.0.0.1:5984/`
  - `{"couchdb" : "Welcome", "version" : "0.10.1"}`
- curl command returns a JSON string
  - Can parse returned string into native object, data structure, or use *as is*
- Illustrates basic CouchDB interaction:
  - Make HTTP request
  - Get HTTP response containing JSON string

# curl: Get list of DBs

- The command:
  - `$ curl -X GET http://127.0.0.1:5984/_all_dbs`
- will yield:
  - `[]`
- Quick aside:
  - Shows functional roots
  - Square brackets `[]` represent ordered lists

# curl: Create a DB

- `$ curl -X PUT http://127.0.0.1:5984/cricket`
- `{"ok" : true}`
- *{}* represent key/value dictionaries, keys must be quoted strings, values can be: strings, numbers, booleans, lists, key/value dictionaries
- `$ curl -X GET http://127.0.0.1:5984/_all_dbs`
- `["cricket"]`

# curl:Another DB

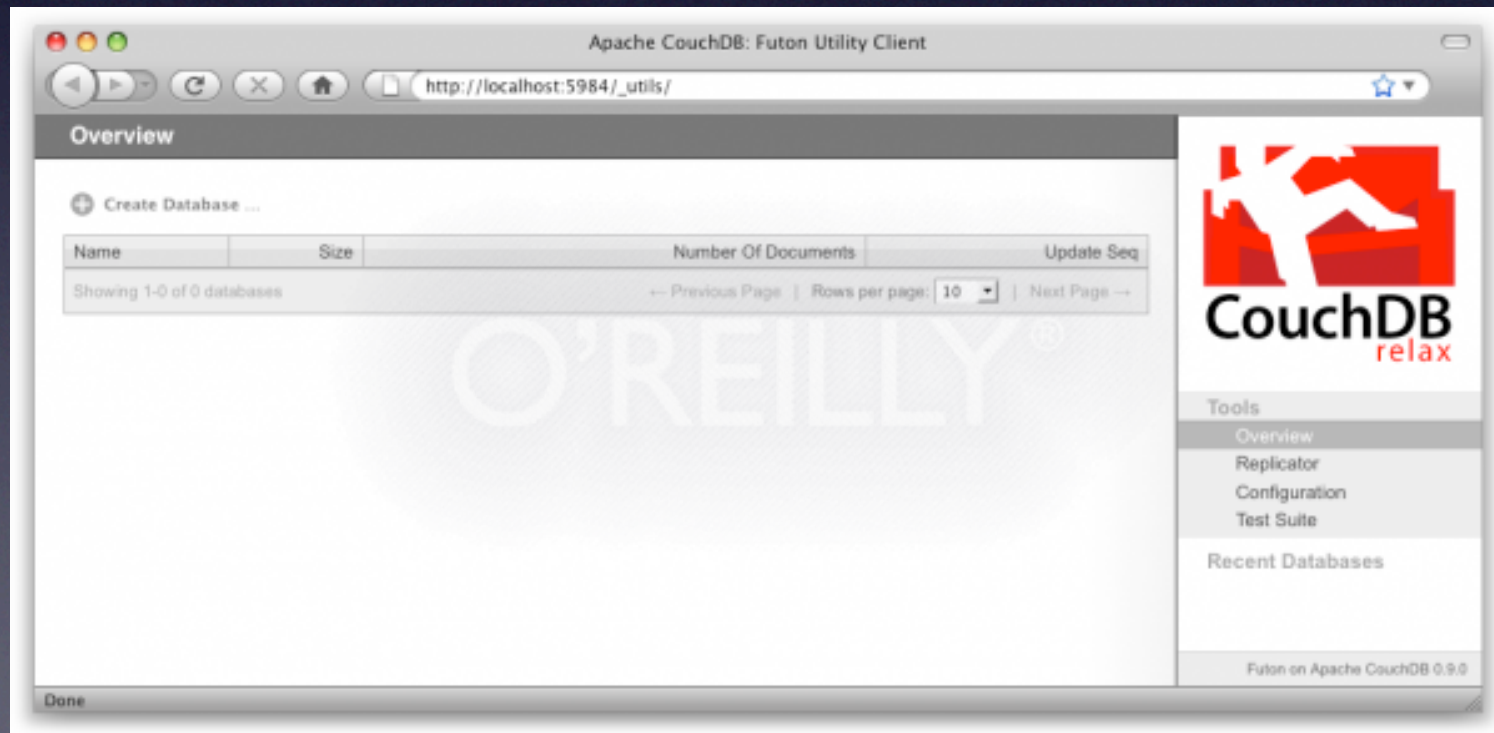
- `$ curl -X PUT http://127.0.0.1:5984/cricket`
- `{“error” :“file_exists”,“reason” :“The database could not be created, the file already exists.”}`
- `$ curl -X PUT http://127.0.0.1:5984/fishing`
- `{“ok” : true}`
- `$ curl -X GET http://127.0.0.1:5984/_all_dbs`
- `[“cricket”,“fishing”]`

# curl: delete a DB

- `$ curl -X DELETE http://127.0.0.1:5984/cricket`
- `{"ok" : true}`
- `$ curl -X GET http://127.0.0.1:5984/_all_dbs`
- `["fishing"]`

# Futon

- In your browser of choice, navigate to:
  - 127.0.0.1:5984/\_utils/
  - *et voila:*



Apache CouchDB: Futon Utility Client

http://localhost:5984/\_utils/


Overview Test Suite

Run All Reload

Note: Each of the tests will block the browser. If the connection to your CouchDB server is slow, running the tests will take some time, and you'll not be able to do much with your browser while a test is being executed.

Name	Status	Elapsed Time	Details
basics	✓ success	629ms	
conflicts	✓ success	141ms	
recreate_doc	✓ success	893ms	
copy_move_doc	✓ success	223ms	
uuids	✓ success	97ms	
bulk_docs	✓ success	179ms	
lots_of_docs	⊗ running...		
reduce	not run		
reduce_false	not run		
multiple_rows	not run		
large_docs	not run		
utf8	not run		
attachments	not run		
content_negotiation	not run		
design_docs	not run		
view_collation	not run		
view_conflicts	not run		
view_errors	not run		
view_include_docs	not run		

Done



Tools

- Overview
- Replicator
- Configuration
- Test Suite

Recent Databases

Futon on Apache CouchDB 0.9.0

# Futon: Test Suite

# Futon: Create a DB

- Now we have a GUI, so:
  - Hit the “Create Database” button
  - Enter a name (hello-world might be *apropo*)

+ Create Document ... ⓘ Compact Database ✕ Delete Database Select view: All documents

Key	Value
Showing 0--1 of 0 rows	



**CouchDB**  
relax

Tools

- Overview
- Replicator
- Configuration
- Test Suite

Recent Databases

hello-world

# Futon: Empty DB

# Futon: Creating Documents

- Hit the “Create Document” button
- CouchDB will assign default `_id` and `_rev` fields
  - NB. When using CouchDB in anger you might want to start setting the IDs yourself
- Hit the “Add Field” button & enter hello followed by `<enter>`

Apache CouchDB: Futon Utility Client

http://localhost:5984/\_utils/

Overview > hello-world > 5467df87655135ae07f093fba0335c9e

Save Document Add Field Upload Attachment Delete Document

Field	Value
_id	"5467df87655135ae07f093fba0335c9e"
_rev	"1680441004"
hello	"world"

Showing revision 3 of 3

Tools

- Overview
- Replicator
- Configuration
- Test Suite

Recent Databases

- hello-world

Futon on Apache CouchDB 0.9.0

Done

# Futon: Hello World Document

Apache CouchDB: Futon Utility Client

http://localhost:5984/\_utils/

Overview > hello-world > 5467df87655135ae07f093fba0335c9e

Save Document Add Field Upload Attachment Delete Document

Fields Source

Source

```
{
  "_id": "5467df87655135ae07f093fba0335c9e",
  "_rev": "1680441004",
  "hello": "world"
}
```

Showing revision 3 of 3

← Previous Version | Next Version →



CouchDB  
relax

Tools

- Overview
- Replicator
- Configuration
- Test Suite

Recent Databases

- hello-world

Futon on Apache CouchDB 0.9.0

Done

Futon: Hello World Document src

# Replication

- Fully updateable
- Occasionally connected replicas
  - Work with offline data

# Futon: Replication

- Can trigger replication between, e.g. two local DBs, a local and a remote DB, or two remote DBs
- Futon makes this easy:
  - Create an empty (target) DB
  - Click “replicator” option in Futon sidebar
  - Select source and target
  - Click **Replicate** button

# MapReduce

- Programming model & associated implementation for processing & generating large datasets
- 2 step: (1) Map function (2) Reduce function
- (1) Specify a map function that processes key/value pairs...
  - ...generating a set of intermediate key/value pairs
- (2) *Optionally* specify a reduce function that merges intermediate key/value pairs associated with the same intermediate key
- Inherently distributable: Map step splits input data set into smaller sub-problems that are distributed to worker nodes (which can in turn map their input data in a recursive fashion)
- <http://labs.google.com/papers/mapreduce.html>

# Views

- Similar to RDBMS views
- read-only, exists independent of the data Static queries of dynamic data.
- Map/Reduce model
- On-demand, incrementally updated indexes View operations: key match, range, limit, etc

# A Simple Example

CRM Database Example contact document:

```
{  
  type: "contact",  
  firstname: "Simon",  
  lastname: "Wells",  
  company: "UoD",  
  ...  
}
```

# A View Function

View of contact documents keyed by lastname:

```
function (doc) {  
  if (doc.type == "contact")  
    map(doc.lastname, doc.firstname);  
}
```

The view functions are stored in special “design documents”, views results accessible via HTTP GET.

# Results

```
GET /db/_view/contacts/by_last?key="Wells"
```

## Result:

```
{ "total_rows": 3,
  "rows": [
    { "id": "2DA92AAA628CB4156134F36927CF4876", "key": "Wells",
      "value": "Simon" },
    { "id": "4156134F36927CF48762DA92AAA628CB", "key": "Wells",
      "value": "Stephen" },
    { "id": "4F36927CF48762DA92AAA628CB415613", "key": "Wells",
      "value": "Thomas" }
  ] }
```

# Show Functions

- Generating HTML (& other formats) from Documents

# CouchApp

- Suite of scripts to ease development and uploading of apps to CouchDB
- Write parts of app offline in a structured fashion
- Use scripts to assemble the individual parts into a CouchDB document and upload to specified location

# Erlang/OTP Primer

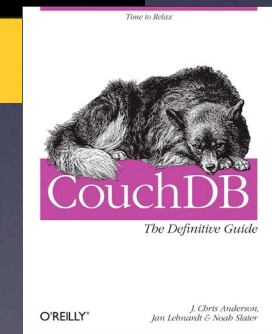
- <http://www.erlang.org/>

# JSON Primer

- <http://www.json.org/>

# Resources

- Project Homepage: <http://couchdb.apache.org/>
- CouchDB Development: <http://www.couch.io/>
- 2 Books available now:
  - Beginning CouchDB:
  - CouchDB: The Definitive Guide
    - More coming soon...



# Summary

- CouchDB-in-a-slide
- History
- Overview
- Document Orientation
- RESTful API
- Using CouchDB with curl
- Using CouchDB with Futon
- Views
- Some background
- Some resources