

# ドキュメント指向データベース CouchDB

株式会社リコー  
グループ技術開発本部  
山本陽平

諸君、私は  
CouchDB が好きだ

- 諸君、私は CouchDB が好きだ
- 諸君、私は CouchDB が好きだ
- 諸君、私は CouchDB が大好きだ

# アンケート

- CouchDBを知っている
- CouchDBをインストールした
- CouchDBを使ってみた
- CouchDBのソースを読んだ
- CouchDBを作っている(コミッタだ)

- ドキュメント指向が好きだ
- Erlangが好きだ
- スキーマレスが好きだ
- Map/Reduceが好きだ
- 半構造データが好きだ

# CouchDB のリソース

- データベース
  - [http://couchdb:5984/{db\\_name}](http://couchdb:5984/{db_name})
  - 0個以上のドキュメントを格納
  - 実体は .couch ファイル
- ドキュメント
  - [http://couchdb:5984/{db\\_name}/{doc\\_id}](http://couchdb:5984/{db_name}/{doc_id})
  - JSON データ
- デザインドキュメント
  - [http://couchdb:5984/{db\\_name}/\\_design/{design\\_doc\\_id}](http://couchdb:5984/{db_name}/_design/{design_doc_id})
  - View を定義する
- ビュー
  - [http://couchdb:5984/{db\\_name}/\\_view/{design\\_doc\\_id}/{view\\_name}](http://couchdb:5984/{db_name}/_view/{design_doc_id}/{view_name})
  - 検索結果リソース

# ドキュメントの例

GET /people/yohei HTTP/1.1

Host: couchdb:5984

HTTP/1.1 200 OK

Content-Type: application/json

ETag: "492284012"

Content-Length: xxx

```
{
  "_id": "yohei",
  "_rev": "492284012",
  "name": "YAMAMOTO Yohei",
  "age": 32,
  "lang": ["java", "ruby", "erlang"]
}
```

# ビューの例

[http://couchdb:5984/people/\\_design/lang](http://couchdb:5984/people/_design/lang)

```
{
  "_id": "_design/lang",
  "language": "javascript",
  "views": {
    "by_lang": {
      "map": "function(doc) {for (i in doc.lang) emit(doc.lang[i], doc._id)}"
    }
  }
}
```

A: [http://couchdb:5984/people/\\_view/lang/by\\_lang](http://couchdb:5984/people/_view/lang/by_lang)

B: [http://couchdb:5984/people/\\_view/lang/by\\_lang?key="ruby"](http://couchdb:5984/people/_view/lang/by_lang?key='ruby')

A

java	yohei
ruby	yohei
erlang	yohei
ruby	foo

B

ruby	yohei
ruby	foo

- Apache Incubator プロジェクトで
- couchdb-devで
- couchdb-userで
- Couchdb Wikiで
- Erlang 分散システム勉強会で

- この地上に存在するありとあらゆる CouchDB が大好きだ
- スキーマレスでデータが突っ込めるのが好きだ
- 既存のドキュメントに新しいプロパティを追加したときなど心がおどる

# スキーマレスデータベース

- 保存するデータにスキーマがない
- (ほぼ)あらゆる JSON ドキュメントを格納可能
- ドキュメント(JSON)のデータ構造は自由に変更可能

- RESTful API で JSON 表現がやり取りされているのが好きだ
- 言語バインディングの存在を忘れていたことに気づいたときなど胸がすくような気持ちだった

# RESTful API

- データベースの作成
  - PUT /people → 201 Created
- ドキュメントの作成
  - PUT /people/yohei → 201 Created
  - POST /people → 201 Created
- ドキュメントの更新
  - PUT /people/yohei → 200 OK
- ドキュメントの削除
  - DELETE /people/yohei → 200 OK
- ほぼすべてのプログラミング言語からアクセス可能
  - コネクタや言語バインディングが不要

- データが分散して保管されている  
感じが好きだ
- 複数台のマシン間でパーシャル  
レプリケーションが行われている  
時など感動すらおぼえる

# 分散データベース

- CouchDB はピアベースの分散DBMS
- データベース単位で双方向レプリケーションを行う
  - 実は JS でフィルタを書けば部分的レプリケーションも行える(らしい)
- コンフリクト解決の戦略
  - データベースの圧縮が行われるまで、すべての変更リビジョンを保管
  - コンフリクトは異常状態ではなく通常状態
  - 手動、またはコンフリクト解決エージェントが解決する

- Erlang OTP で動いているときなどもうたまたまらない

# Erlang

- CouchDB は Erlang で書かれている
  - 6635行
  - Web サーバは MochiWeb を利用
- Stroke DB (ruby) は 7000行以上
  - 単純には比べられないが、ある種の問題については ruby よりも Erlang の方が短く書ける

- JavaScriptで map と reduce を書いたときは最高だ

# Map/Reduce

- View は map 関数だけでなく、reduce 関数も持てる

View:

```
{  
  "average": {  
    "map": "function(doc) { emit(doc._id, doc.age); }",  
    "reduce": "function(key, values) { return sum(values) / values.length; }"  
  }  
}
```

Result:

```
{"rows": [{"key": null, "value": 2.96666666666666678509e+01} ]}
```

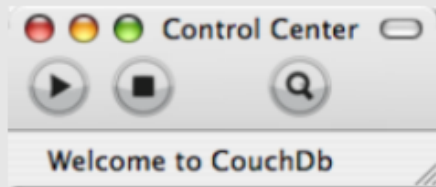
- RDBでは関係に無理やりタグを突っ込んでいるのを
- JSON ドキュメントにプロパティ一個追加するだけで対応する時など絶頂すら覚える

- 分散データベースとして設計されている CouchDBが好きだ
- 単なるモバイルデータベースと勘違いされているときはとてもとても悲しいものだ
- MVCCでバージョン管理されているのが好きだ
- ロックがない不完全なDBMSと誤解されるのは屈辱の極みだ

# 誤解される CouchDB

## Web2.0時代のニュータイプDB「CouchDb」

[GPL] [ GUI] [ JavaScript] [ Mac OSX] [ Web] [ Web API/Mashup] [ Windows] [ オープンソース] [ データベース]



Web2.0時代(?)の技術要素の一つにWeb APIがある。また、各種アプリケーションがWeb化され、ブラウザ内で動作するようになっている。

しかし、これらに必須なのがインターネットだ。ネットワークに繋がっていないければ利用できない。その限界を超えるかも知れないソフトウェアがこれだ。

今回紹介するオープンソースソフトウェアはCouchDb、新しい形式のドキュメントデータベースだ。

CouchDbは簡単に言うとRESTを通じたWeb API経由でデータ取得ができるデータベースだ。結果はJSONで受け取ることができる。そのため、PHP/Ruby/Java/LotusScript/ひなど、言語を問わず広く利用できる。

興味深いのは、このCouchDb自体は各クライアントに入れて動作させるという方法だ。この場合、オフライン状態でも利用でき、オンラインになると同期処理をすることができる。

どういった用途に向いているのか、というのはいくつかは言えないが公式サイトではバグトラッキングシステムや掲示板のデモがある。また、CRM/アドレス帳/Wiki/ToDoなどを例として挙げている。

JSONが前提状態と言うのが興味深い。リレーショナルでもオブジェクトでもない、ドキュメントデータベースと言う新しい概念、使ってみたくはないだろうか。



CouchDb Project Website

- 諸君 私はCouchDBを 研ぎ澄まされた剣の様なCouchDBを望んでいる
- 諸君 私に付き従うCouchDB好きの諸君 君たちは一体何を望んでいる？
- 更なるCouchDBを望むか
- 情け容赦のない CouchDBを望むか？
- 分散し、スキーマフリーで、ドキュメント指向の限りをつくしたようなCouchDBを望むか？

# ドキュメント指向データベース

- != OODBMS
- OODBMS は「オブジェクト」を格納
- CouchDB は「ドキュメント」を格納
- たとえば Lotus Notes/Domino はドキュメント指向
- いわゆる XML データベース(スキーマレスのもの)もドキュメント指向なものが多い

• CouchDB ! !

• CouchDB ! !

• CouchDB ! !

- よろしい ならばCouchDBだ
- だが、RDBMS 全盛の世の中で耐え続けて来た我々には
- ただのCouchDBではもはや足りない！！
- 大CouchDBを！！
- 一心不乱の大CouchDBを！！

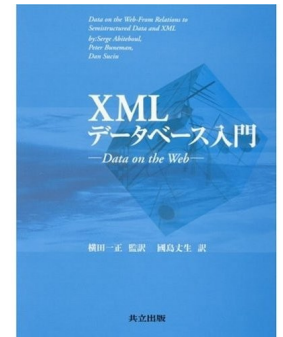
- 我々はわずかに小数
- RDBMSに比べれば物の数ではない
- だが諸君は一騎当千のErlangプログラマだと私は信じている
- ならば我らは諸君と私で総兵力3万と1人の分散プログラミング集団となる
- 我らを忘却の彼方へと追いやり眠りこけている、RDBMSを叩きのめそう
- 眼(まなこ)をあけて思い出させよう

- 連中にデータを自由に保管する楽しみを思い出させてやる
- 連中に表形式以外のデータ構造を思い出させてやる

- CouchDBには奴らの哲学では思いもよらないスキーマレスによる自由なデータ構造がある事を思い出させてやる
- 30人のErlangプログラマの集団で世界をCouchDBとMochiWebで埋め尽くしてやる

# 半構造データベース

- Web 時代のデータ == 半構造データ(Semi-structured data)
  - スキーマレス
  - 自己記述的
- 10年前くらいからの研究
  - RDB には入れにくい、不確定な構造を持つデータのためのデータモデル
- 無理やり RDB に入れているデータは、CouchDB などの半構造データベースで管理する方が良い(場合もある)
  - タグとか



- 目標 Web データ
- Web データ分散保管作戦
- 状況を開始せよ
  
- いくぞ 諸君

おわり