Ch.5 Design Documents

CouchDB Definitive Guide 勉強会 #2

RelaxCafe@CouchDB Break.2

まえだこうへい
id:mkouhei(CouchDB-JP)

2009 年 10 月 16 日

アジェンダ

- はじめに
- 文書設計って何よ？
- 問い合わせしましょ。
- アプリは文書だ！
- 基礎は大事です。
- 基礎以外のその他って何だ？

はじめに

デザインドキュメントとは？

- アプリのコードも含む CouchDB の特殊なドキュメント。
- アプリの API を高度に構造化できる。

以上。

では話にならんので、続けます。

文書設計ってなによ？

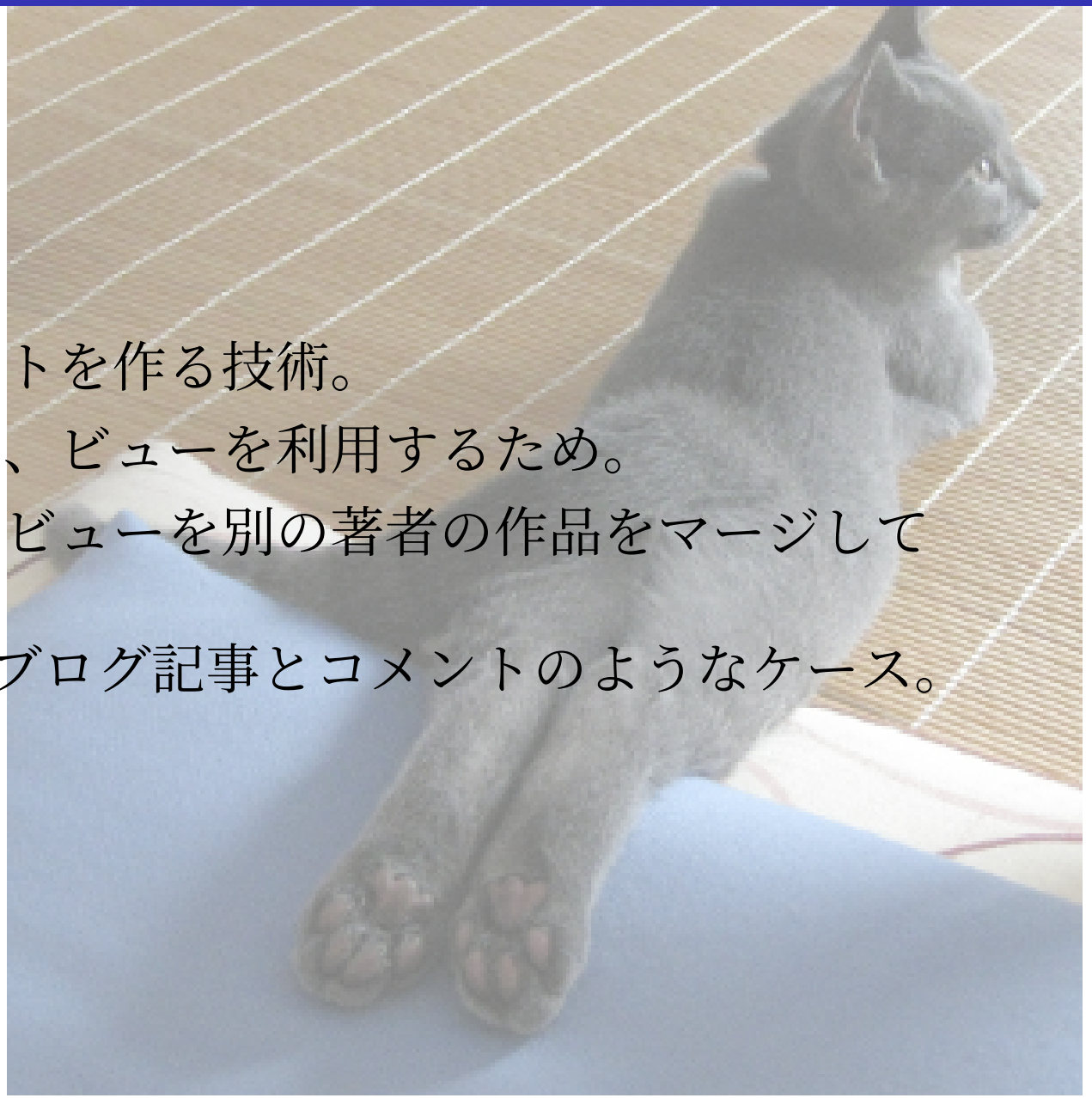
なぜドキュメントモデリングするのか？

ドキュメントと呼ぶものは２種類考えられる。

- ワードプロセッサ
- ユーザプロファイル

これらでデータのソートを行うには、“痛い正規化” が必要。
簡単に行うためには何か工夫が必要。

ドキュメントモデリングとはビューなり。



- 仮想ドキュメントを作る技術。
- データを照合し、ビューを利用するため。
- 例：提供されるビューを別の著者の作品をマージして作成するなど。
  1クエリ内でのブログ記事とコメントのようなケース。

ちなみに別ドキュメントでの属性保存は？

そんなの、ナンセンスだ！

使い途の例

例えば、イベントログ。
最低限の検証が必要なイベントでのデータ保存時に
ユーザの操作を記録する。

- ユーザの入力を信頼していないようなケース
- 非同期のジョブをトリガーとする必要があるケース

問い合わせしましょ。

クエリサーバーとは？

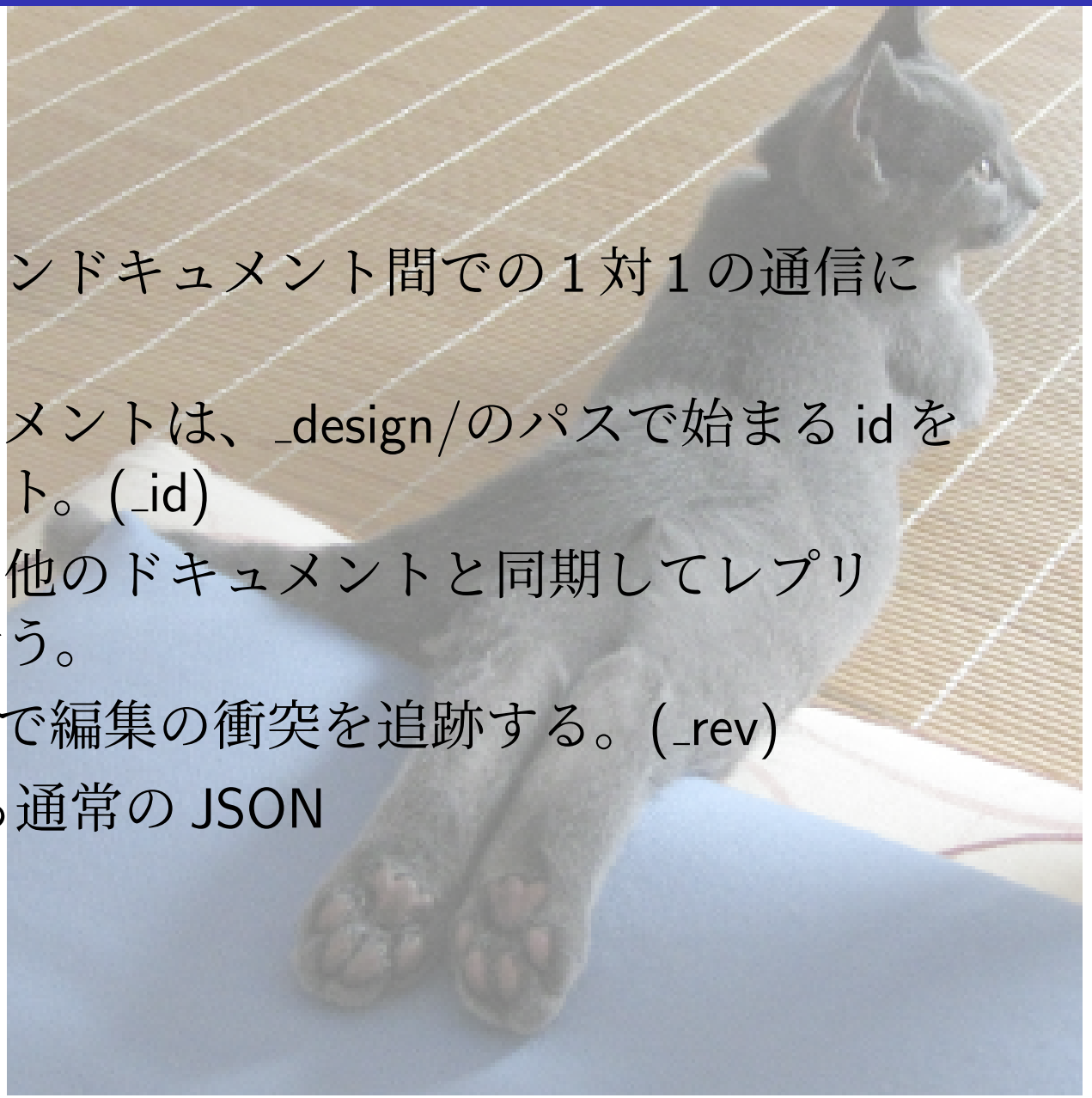
- デフォルトのクエリサーバーは JavaScript で実装。
- デザインドキュメントの機能は、ほぼ全ての言語でビューサーバとして利用できる。
- JSON コマンドが使えれば実装できるよ。

クエリサーバで利用できる機能

- 既存機能
  - MapReduce ビュー
  - 更新のチェック機能
  - show や list 変換
- ロードマップ上利用できる機能
  - レプリケーションフィルター
  - 非 JSON 入力データ解析用の更新処理
  - アプリケーション URI を生成する上書き処理

アプリは文書だ！

CouchDB のアーキテクチャ

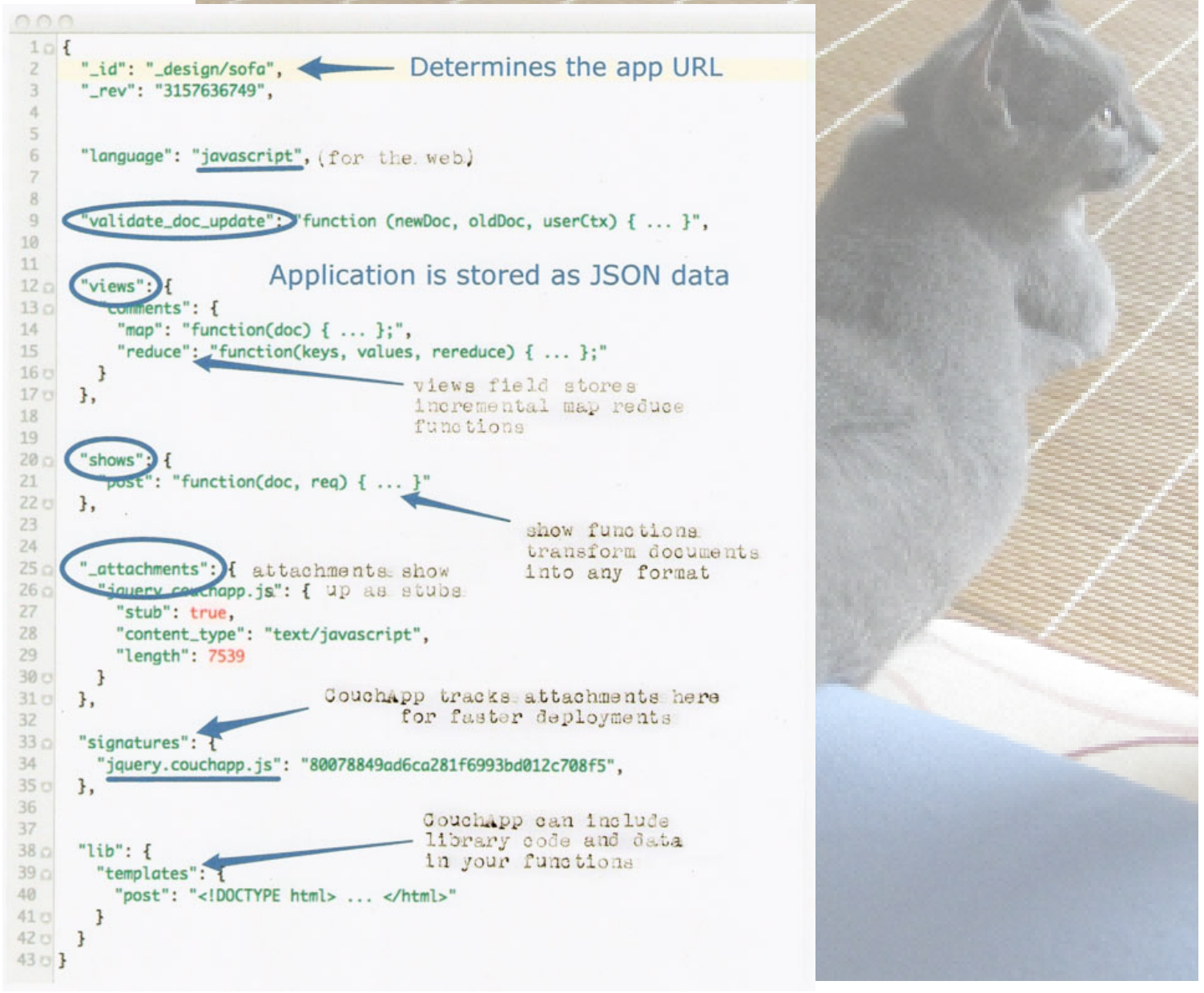


- アプリとデザインドキュメント間での 1 対 1 の通信に向いている。
- デザインドキュメントは、_design/のパスで始まる id を持つドキュメント。(_id)
- データベースの他のドキュメントと同期してレプリケーションを行う。
- _rev パラメータで編集の衝突を追跡する。(_rev)
- _design で始まる通常の JSON

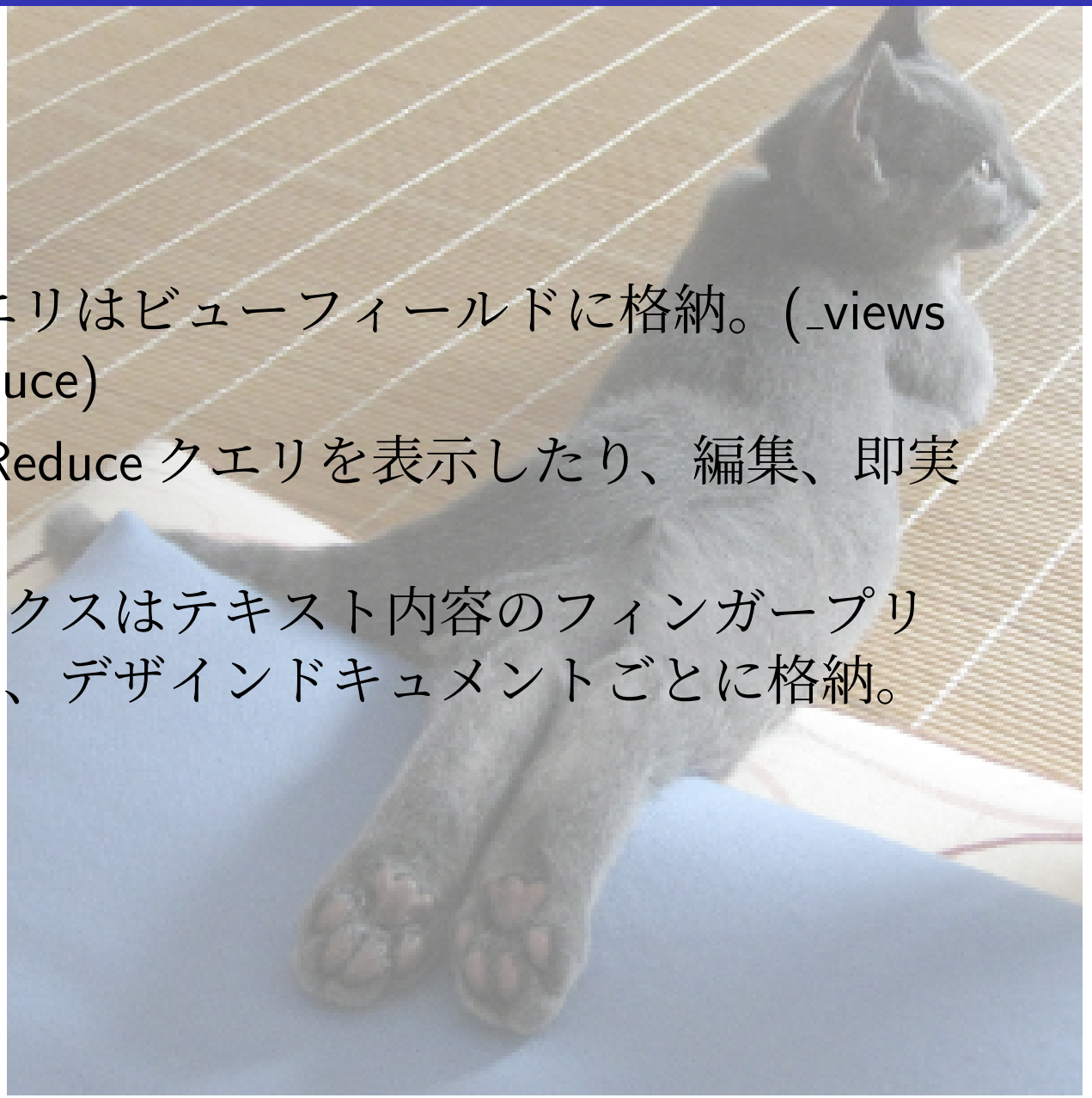
CouchDB のビューとアプリの関係

- アプリの静的 HTML ページは、デザインドキュメントを添付ファイル。(_attachments)
- ビューと検証は、デザインドキュメントの JSON ボディ部に含まれる。(views，validate_doc_update)

解体新書

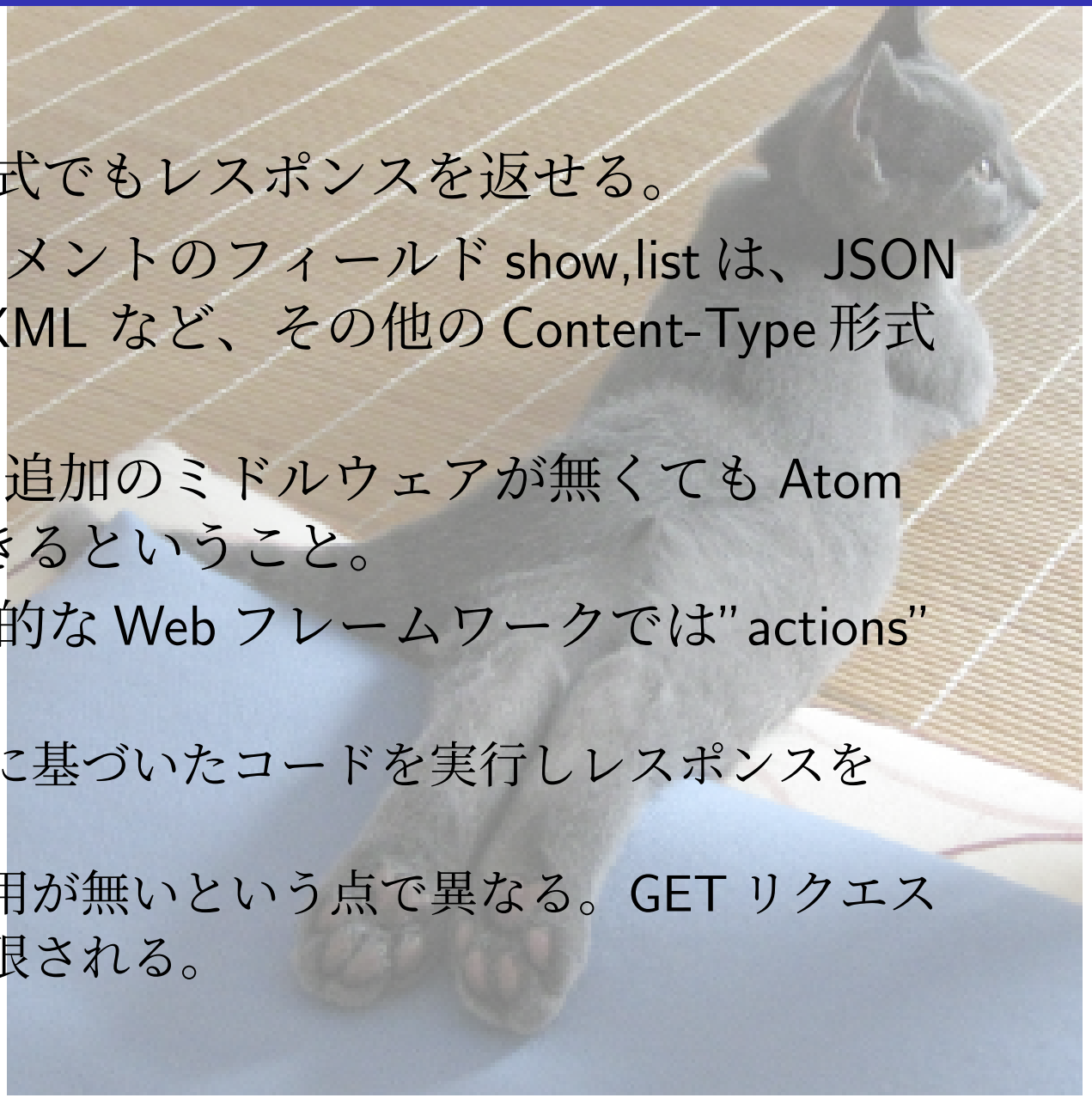

{
"_id": "_design/sofa",
"_rev": "3157636749",
"language": "javascript", (for the web)
"validate_doc_update": "function (newDoc, oldDoc, userCtx) { ... }",
"views": {
"comments": {
"map": "function(doc) { ... };",
"reduce": "function(keys, values, rereduce) { ... };"
}
},
"shows": {
"post": "function(doc, req) { ... }"
},
"_attachments": { attachments show
"jquery.couchapp.js": { up as stubs
"stub": true,
"content_type": "text/javascript",
"length": 7539
}
},
"signatures": {
"jquery.couchapp.js": "80078849ad6ca281f6993bd012c708f5",
},
"lib": {
"templates": {
"post": "<!DOCTYPE html> ... </html>"
}
}
}
Determines the app URL
Application is stored as JSON data
views field stores incremental map reduce functions
show functions transform documents into any format
CouchApp tracks attachments here for faster deployments
CouchApp can include library code and data in your functions

CouchDB は布団で MapReduce の夢を見るか？



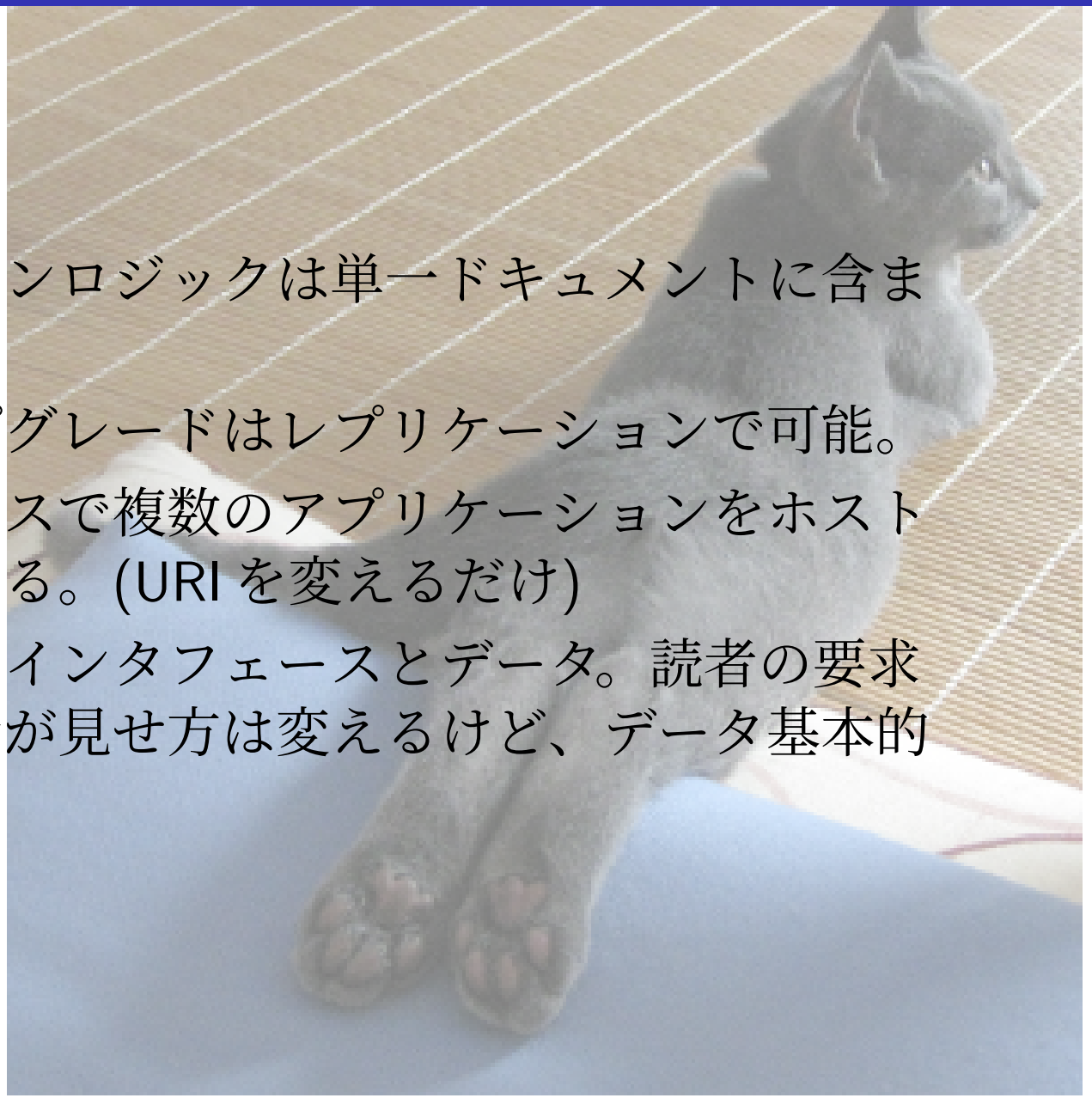
- MapReduce クエリはビューフィールドに格納。(_views の下の map, reduce)
- Futon で、MapReduce クエリを表示したり、編集、即実行できる。
- ビューインデックスはテキスト内容のフィンガープリントで紐付けし、デザインドキュメントごとに格納。(signatures)

レスポンスだ。



- JSON 以外の形式でもレスポンスを返せる。
- デザインドキュメントのフィールド show,list は、JSON から HTML や XML など、その他の Content-Type 形式に変換できる。
- ブログの例なら追加のミドルウェアが無くても Atom feeds を提供できるということ。
- show,list は伝統的な Web フレームワークでは"actions"に似ている。
  - リクエストに基づいたコードを実行しレスポンスを返す。
  - しかし副作用が無いという点で異なる。GET リクエストだけに制限される。

アプリも一緒。



- アプリケーションロジックは単一ドキュメントに含まれている。
- コードのアップグレードはレプリケーションで可能。
- 単一データベースで複数のアプリケーションをホストすることができる。(URI を変えるだけ)
- 例えば、新聞のインタフェースとデータ。読者の要求に応じて編集者が見せ方は変えるけど、データ基本的に同じ。

デザインドキュメントは複数扱える。

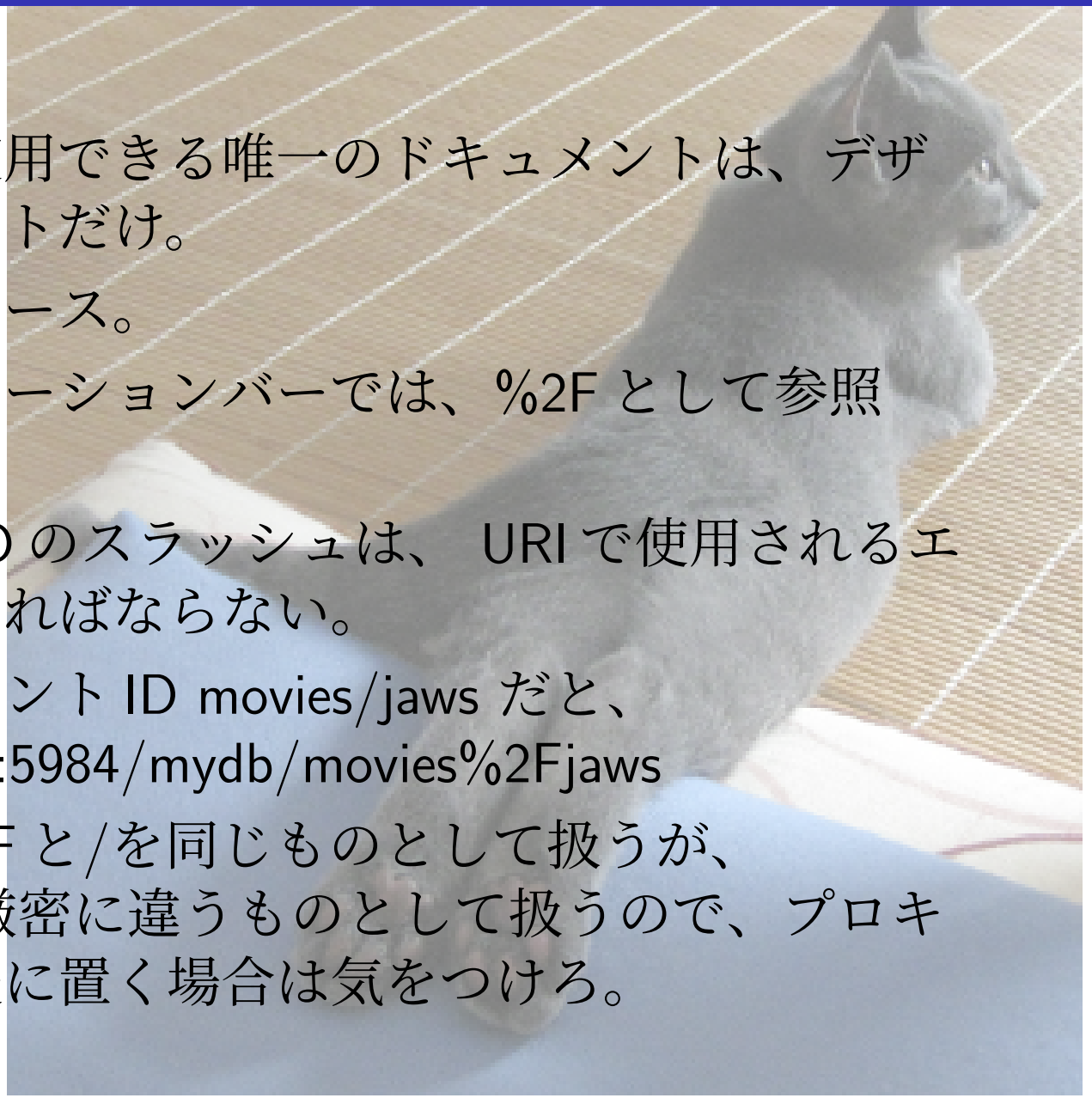
デザインドキュメントの ID を例に挙げると、

- _design/calendar
- _design/contacts
- _design/blog
- _design/admin

デザインドキュメントの JSON データを URI をしていして GET できる。

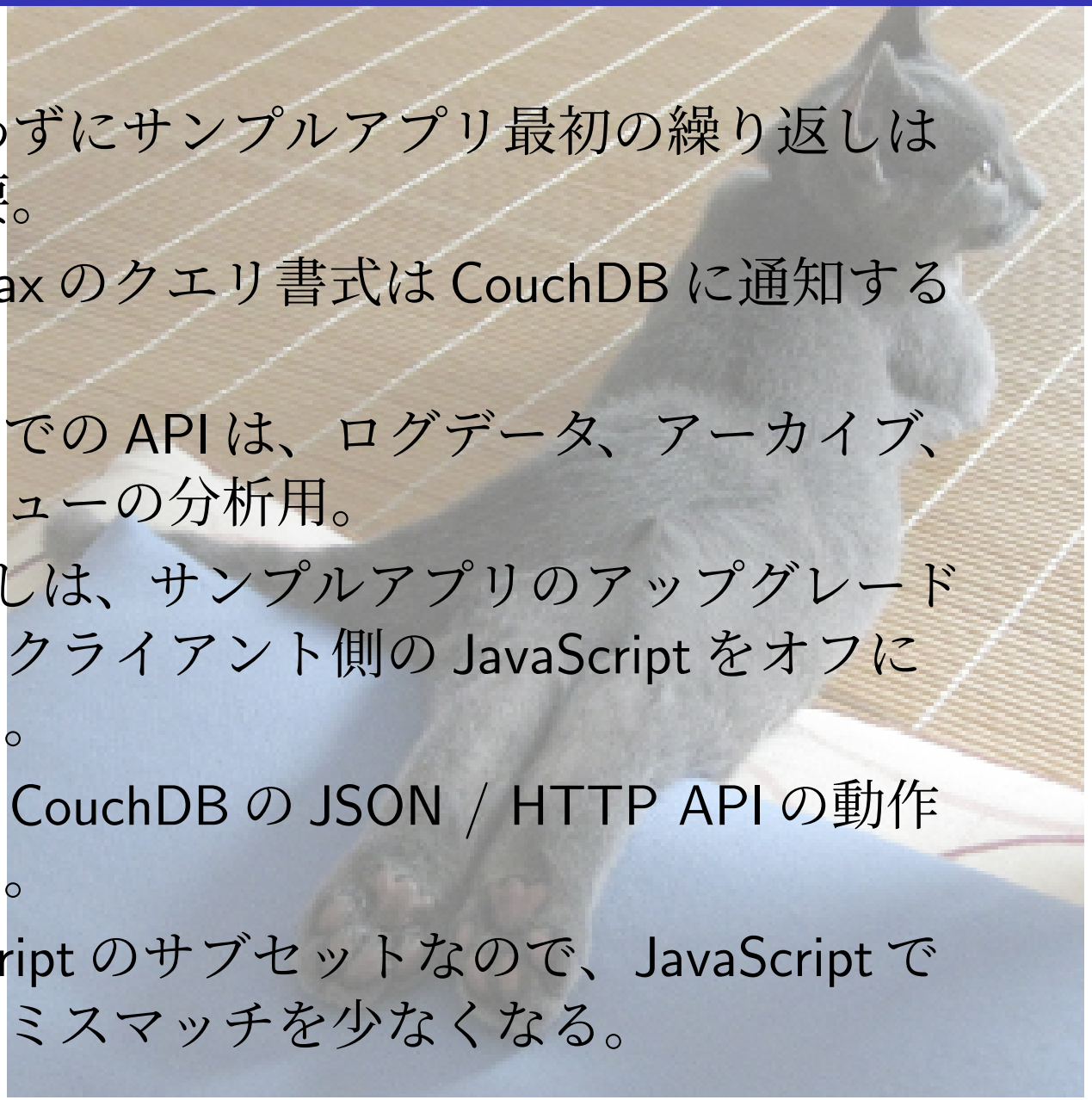
- http://localhost:5984/mydb/_design/calendar
- http://localhost:5984/mydb/_design/contacts
- http://localhost:5984/mydb/_design/blog
- http://localhost:5984/mydb/_design/admin

URI 内の/(スラッシュ)について



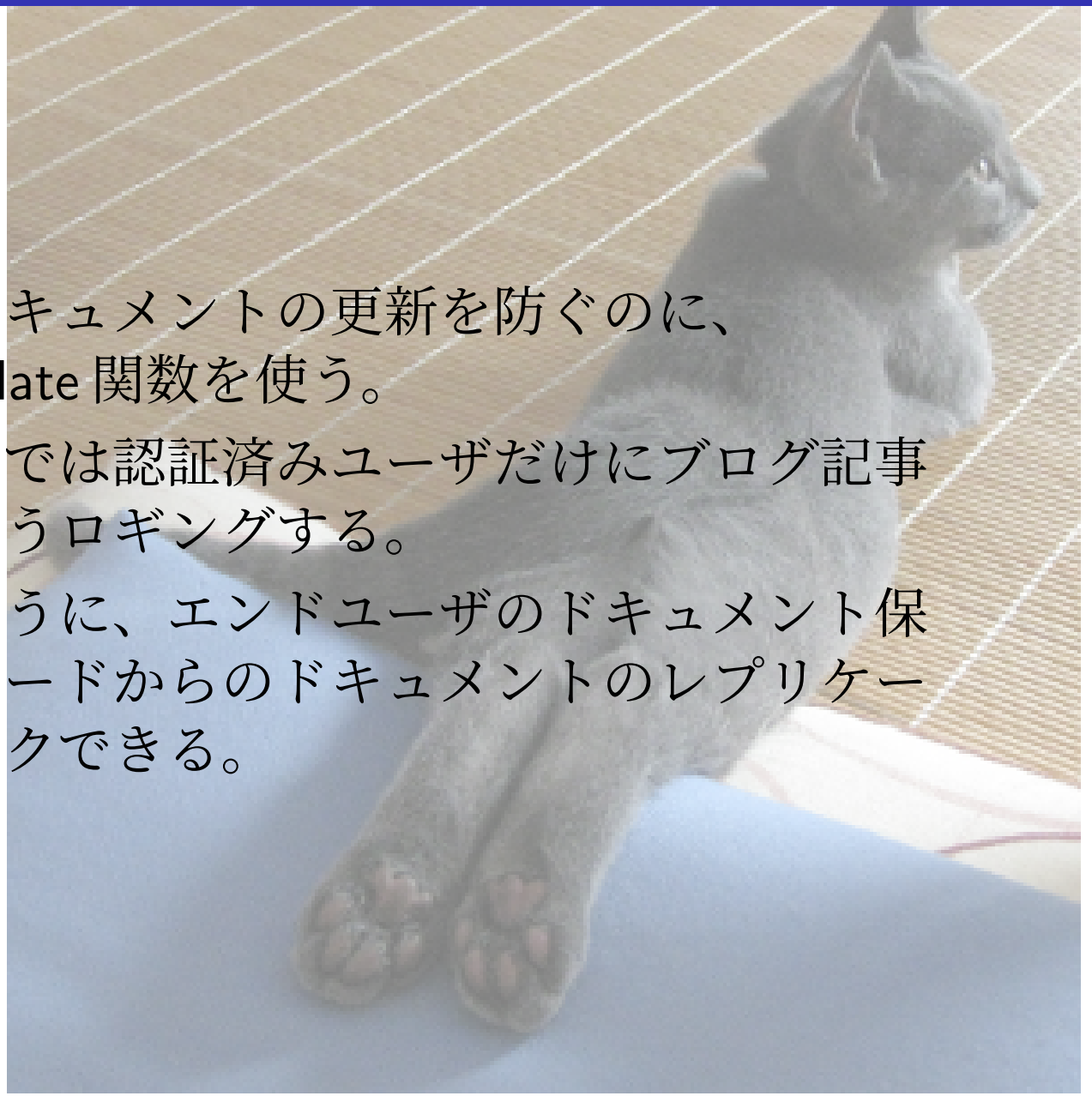
- スラッシュを使用できる唯一のドキュメントは、デザインドキュメントだけ。
- これは特殊なケース。
- ブラウザのロケーションバーでは、%2F として参照する。
- ドキュメント ID のスラッシュは、 URI で使用されるエスケープしなければならない。
- 例えばドキュメント ID movies/jaws だと、http://localhost:5984/mydb/movies%2Fjaws
- Apache では%2F と/を同じものとして扱うが、CouchDB では厳密に違うものとして扱うので、プロキシサーバの背後に置く場合は気をつけろ。

13 日の金曜日



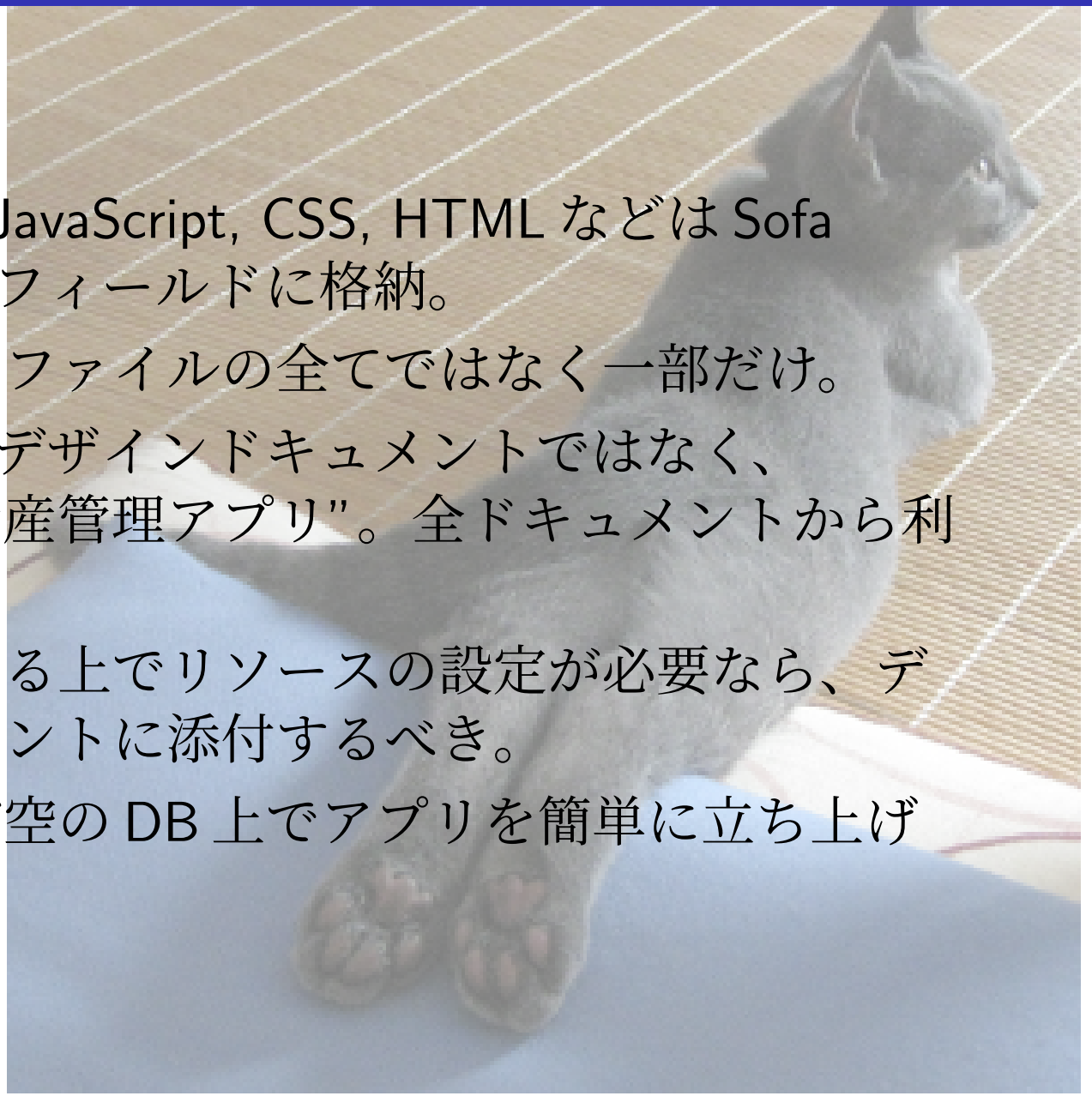
- show, list を使わずにサンプルアプリ最初の繰り返しは show, list は不要。
- JSON API の Ajax のクエリ書式は CouchDB に通知するのによい。
- 最初の繰り返しでの API は、ログデータ、アーカイブ、管理する永続キューの分析用。
- 2 番目の繰り返しは、サンプルアプリのアップグレードの例のように、クライアント側の JavaScript をオフにしても機能する。
- Ajax クエリは、CouchDB の JSON / HTTP API の動作の透過性が高い。
- JSON は JavaScript のサブセットなので、JavaScript でインピーダンスミスマッチを少なくなる。

飲酒チェックだな。



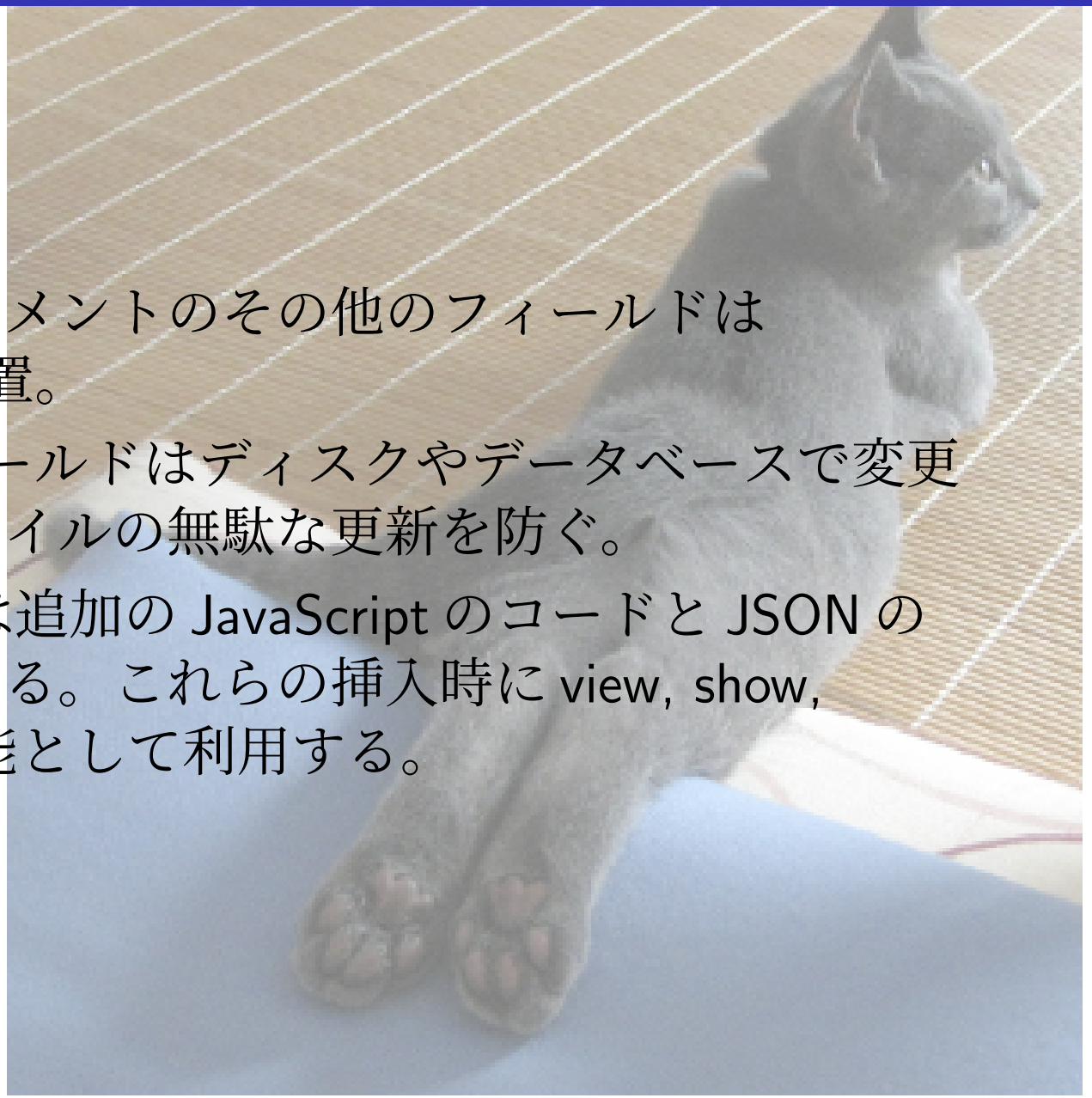
- 無効、不正なドキュメントの更新を防ぐのに、validate_doc_update 関数を使う。
- サンプルアプリでは認証済みユーザだけにブログ記事を投稿できるようロギングする。
- 副作用が無いように、エンドユーザのドキュメント保存以外に、他ノードからのドキュメントのレプリケーションもブロックできる。

添付ファイルの扱い



- raw イメージ、JavaScript, CSS, HTML などは Sofa の_attachments フィールドに格納。
- デフォルトではファイルの全てではなく一部だけ。
- _attachments はデザインドキュメントではなく、CouchDB の" 資産管理アプリ"。全ドキュメントから利用できる。
- アプリを実行する上でリソースの設定が必要なら、デザインドキュメントに添付するべき。
- 新しいユーザが空の DB 上でアプリを簡単に立ち上げられる。

CouchAppの部分。



- デザインドキュメントのその他のフィールドはCouchAppに配置。
- signatures フィールドはディスクやデータベースで変更のない添付ファイルの無駄な更新を防ぐ。
- lib フィールドは追加の JavaScript のコードと JSON のデータを保持する。これらの挿入時に view, show, validation の機能として利用する。

基礎は大事です。

基本的なデザインドキュメント

まずは、次のテキストを mydesign.json として保存してみよう。

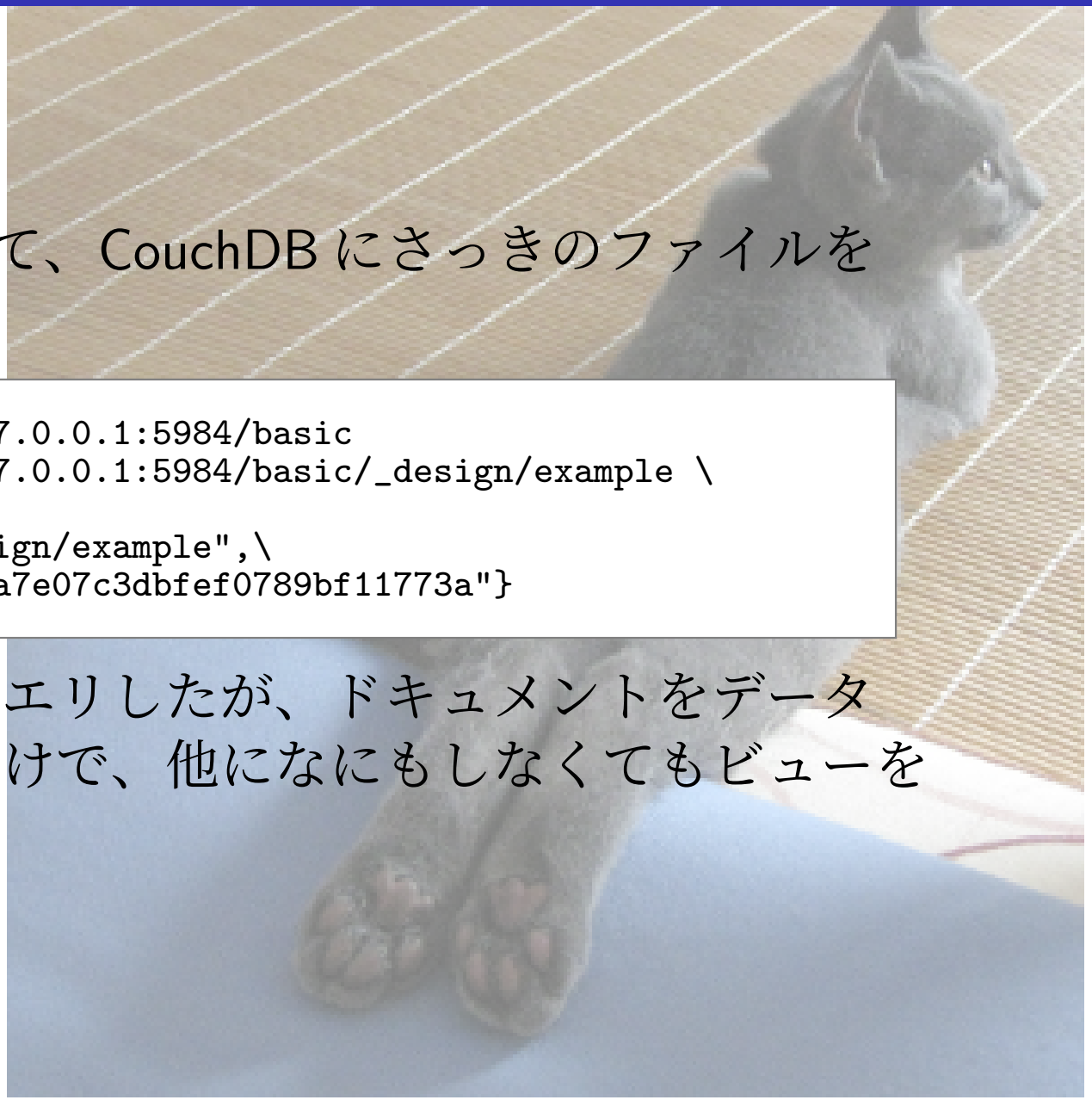
```
{
  "_id" : "_design/example",
  "views" : {
    "foo" : {
      "map" : "function(doc){ emit(doc._id, doc._rev)}"
    }
  }
}
```

基本的なデザインドキュメント

curl コマンドを使って、CouchDB にさっきのファイルを PUT する。

```
curl -X PUT http://127.0.0.1:5984/basic
curl -X PUT http://127.0.0.1:5984/basic/_design/example \
   -d @mydesign.json
{"ok":true,"id":"_design/example",\
   x"rev":"1-230141dfa7e07c3dbfef0789bf11773a"}
```

定義したビューをクエリしたが、ドキュメントをデータベースに追加するだけで、他になにもしなくてもビューを得られる。



基本的なデザインドキュメント

次のコマンドを実行すると、空のドキュメントを追加できる。

```
curl -X POST http://127.0.0.1:5984/basic -d '{}'
```

このビューをクエリするには

```
curl http://127.0.0.1:5984/basic/_design/example/_view/foo
```

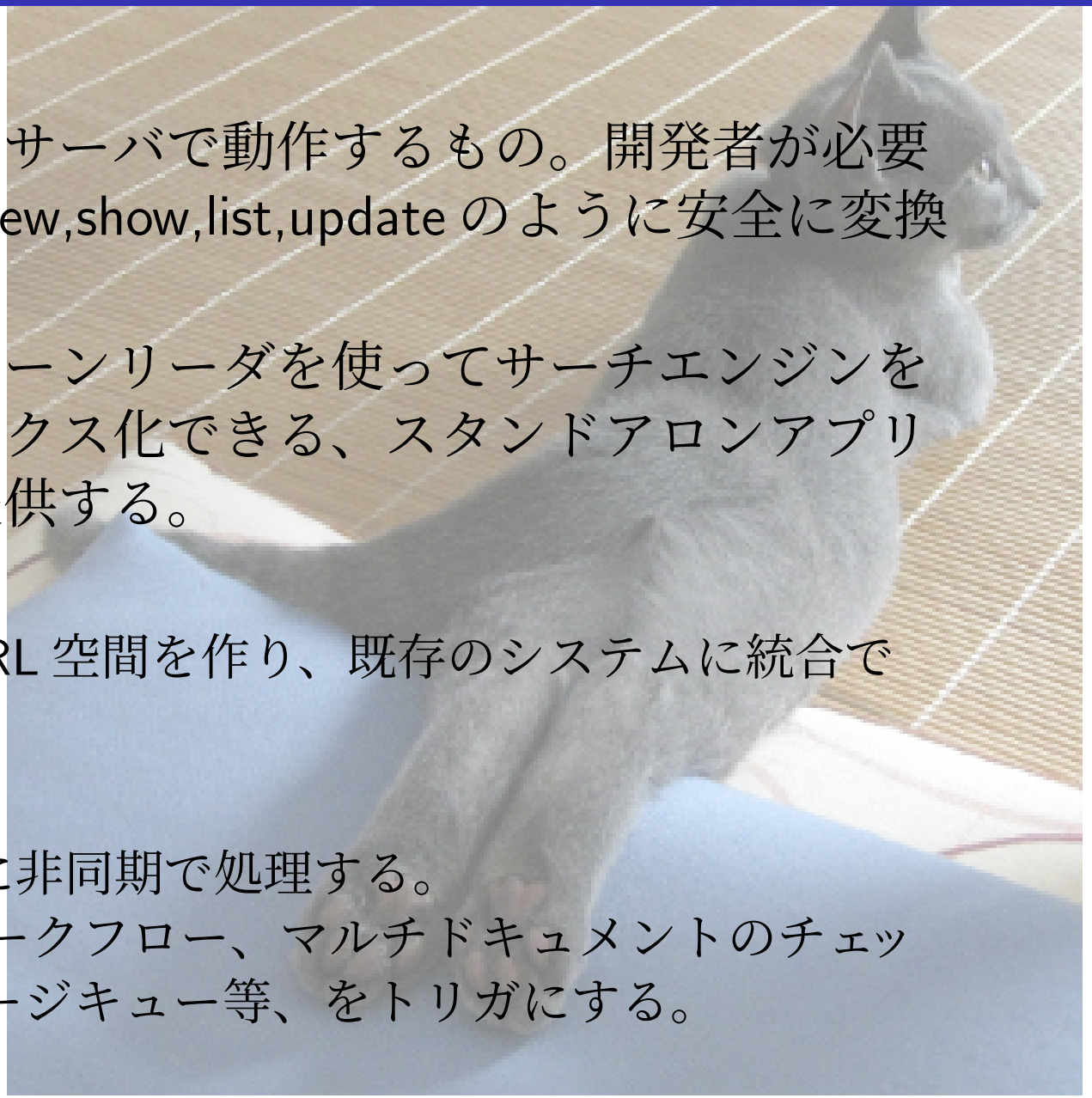
つまり、 データベース内のドキュメントはすべて list できるということ。 (デザインドキュメントは除く)。

他にゃなんかねえのかい？

実装済みの他の機能

- _update
  - 前提：部分的なイベント発生時に、web サービス A が XML-blob を選択した URI に POST する場合。例えば Paypal のインスタント支払い通知。
  - CouchDB に直接 POST を行える。XML をパースし、JSON ドキュメントとして格納できる。
  - 同様に CSV やマルチパート形式など他の形式にも変換できる。
- _filter
  - _filiter は” 第 25 章 Change Notification” までマテ。

未実装の他の機能



- 大目標：アプリサーバで動作するもの。開発者が必要とする機能は view,show,list,update のように安全に変換できること。
- 中目標：スクリーンリーダを使ってサーチエンジンを簡単にインデックス化できる、スタンドアロンアプリの構築手段を提供する。
- rewrite
  - アプリの URL 空間を作り、既存のシステムに統合できる。
- event
  - DB 変更時に非同期で処理する。
  - 例えば、ワークフロー、マルチドキュメントのチェック、メッセージキュー等、をトリガにする。