CubbyでRESTfulなWebアプリを

株式会社ヌーラボ　縣俊貴

自己紹介

- 縣俊貴(id:agt)
- 株式会社ヌーラボ
  - SI - アジャイル開発のヌーラボ
  - Webサービス
    - プロジェクト管理ツールBacklog
- Seasar
  - S2Pager/S2XML-RPC/Cubby
- WEB+DB PRESS
  - 連載：良いコードへの道

アジェンダ

1. 私がCubbyを作った理由
2. CubbyのRESTfulサポート
3. アプリ作成のデモ
4. 今後のロードマップ

私がCubbyを作った理由

2006年7月

Webフレームワーク戦国時代

- Struts
- WebWork2
- S2JSF
- Teeda
- Ruby on Rails
- etc…

自分にとってぴったりのフレームワークがない！

ぴったりな
ものって？

ぴったりなもの

- JSP
  - 嫌われ者だけど、そんなにみんな嫌いなの？
  - HTMLテンプレートって、意外とつらくない？
  - JSP2.0 JSTL/ファンクション/シンプルタグ/タグファイル/
- クールURI
  - 直感的に楽に使いたい
  - 自由度は最大限に
- 設定ファイルレス
  - ただし、わかりにくくなってはだめ

では、
ぴったりなものを
作ろう！

２００６年８月
Cubby開発開始

当初は社内のSI案件をさくさくこなすために作成

今ではいろいろと
おもしろいサービスに
使われはじめています。

http://www.choistudy.jp/

http://tophatenar.com/

http://hatenarmaps.com/

閑話休題

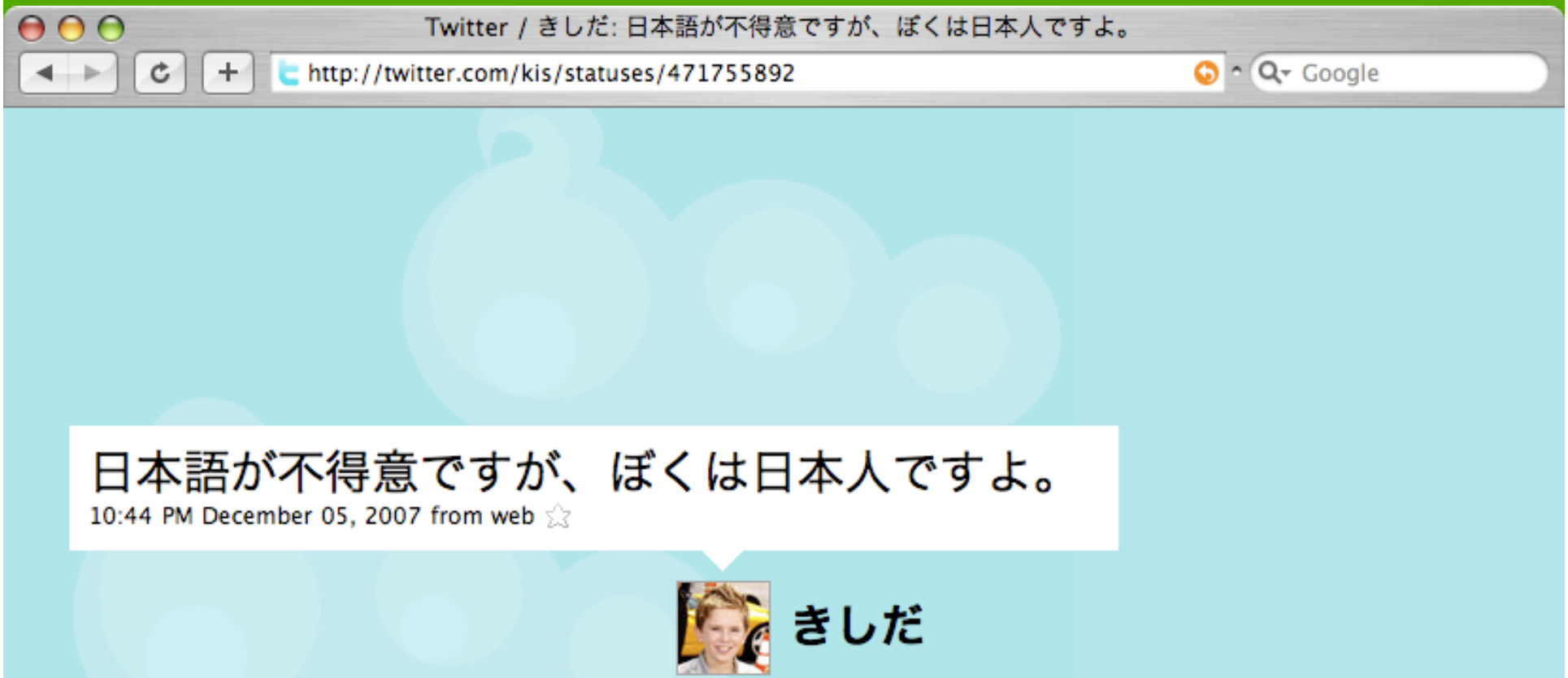
- フレームワークやライブラリ作りは経験しておいて損はないと思います。
  - 抽象化
  - パターン
- みんな俺フレームワークを作ればいいと思います。
  - １人１フレームワーク
  - 似たフレームワークが乱立しても別にいいじゃない
  - 失敗してもいいじゃない、人間だもの。

CubbyのRESTfulサポート

２種類のREST

- APIとしてのREST
  - システム外部公開用のAPI
  - GET/POST/PUT/DELETEによる操作
  - AtomPP/XML/Json/XHTMLなどによるデータ構造
- RESTfulなWebアプリケーション
  - Cool URI
  - GET/POST
  - HTML/XHTMLによるデータ構造

http://twitter.com/kis/statuses/471755892



RESTfulなWebアプリケーションの嬉しさ

- 将来的にバックエンドのシステムが変わっても、URIが変更されない
  - 拡張子なし
- URIがリソースごとに階層化されていて整理されているので、気持ちいい。
  - /{ユーザ名}/{タグ}
  - パッケージ名などと同じ

Cubby以前

- mod_rewrite

```
#httpd.conf
RewriteRule ^\/users\/([0-9]+)? /user.do?userId=$1
```

- URLRewriteFilter

```
<rule>
  <from>users/(.*)$</from>
  <to type="forward">user.do?useId=$1</to>
</rule>
```

Cubby以前

http://example.com/users/agt

・設定ファイルが２つのFWに分散
・論理的なURIが２つあると、大変

Cubbyでは？

http://example.com/users/agt

@Path(“/users/{userId}”)

・設定はアクションクラスに一元化
・処理の近くに設定があるのでわかりやすい

@Path

```
@Path(“todo”)  // TodoActionの場合省略可
public class TodoAction extends Action {
  // /todo/new
  public ActionResult new() { … }
  // /todo/save
  @Path(”save")
  public ActionResult post() { … }
}
```

@Path（パステンプレート）

```
@Path("todo")  // TodoActionの場合省略可
public class TodoAction extends Action {
  public String id;
  // /todo/{id}
  @Path("{id}")
  public ActionResult index() {
    System.out.println(id);
  }
}
```

CubbyでエコURI

- 貴重なURIをエコに利用する仕組み
  - 正規表現
  - 優先度
  - リクエストメソッド
  - Submitされたボタンによる振り分け

正規表現を使った柔軟なURI指定

@Path(”/todo/{id}”) -> [a-z][A-Z][0-9]+
「/todo/0fabd3f」にマッチ

@Path(”/todo/{id,[0-9]+}”)
「/todo/10001」にマッチ

JSR-311の仕様もだいたい同じ！

Path(”/icon/{width,[0-9]+}x
{height, [0-9]+}.{ext,png|jpg}”)
「/icon/100x200.png」にマッチ

優先度

@Path(”/todo/{id}”)
(デフォルト値：
priority=Integer.MAX_VALUE)

@Path(value=”/todo/list”, priority=0)
/todo/listのほうが優先される

リクエストメソッド

```
import static org.seasar.cubby.action.RequestMethod.*;

@Accept(GET)
public ActionResult index() { … }

@Accept(POST)
public ActionResult add() { … }

@Accept(PUT)
public ActionResult update() { … }

@Accept(DELETE)
public ActionResult delete() { … }
```

URIマッピングの確認

- コンソールログ

```
アクションメソッドを登録します [regex=^/todo/$,method=public
    org.seasar.cubby.action.ActionResult
    org.seasar.cubby.examples.todo.action.TodoListAction.index(),uriPar
    ameterNames=[],requestMethod=GET,onSubmit=null,priority=21474
    83647,auto=true]
   …
```

- 管理サーブレット

Cubby Admin Tool

URI: /todo/123 テスト クリア

No	正規表現パターン	HTTPメソッド	アクションメソッド	パスパラメータ	Priority
95	^/todo/([0-9]+)$	GET	org.seasar.cubby.examples.todo.action.TodoAction#show	{id=123}	2147483647
96	^/todo/([0-9]+)$	POST	org.seasar.cubby.examples.todo.action.TodoAction#show	{id=123}	2147483647

Path to Entity

- RESTfulの場合、パステンプレートの変数とエンティティの変換だらけ
- /agata/entries/123
  - name=‘agata’のAccount
  - id=123のEntry
- Conveter
  - リクエストパラメータの変換機構

Converterの作成

```
public AccountConverter extends AbstractConverter {
  public AccountDao accountDao;
  public Class<?> getObjectType() {
     return Account.class;
  }
  public Object convertToObject(Object value, Class<?> objectType,
    ConvertionHelper helper) {
    if (value == null) { return null; }
    String name = String.class.cast(value);
    return accountDao.findByName(name);
  }
  public String convertToString(Object value, ConvertionHelper helper) {
    Account account = Account.class.cast(value);
    return account.getName();
  }
}
```

文字列(agata)
↓変換処理
Entiy(Account)

Entiy(Account)
↓変換処理
文字列(agata)

Converterを使用したAction

- /agata/entries/123

```
public EntryAction extends Action {
 @RequestParameter
  public Entry entry;
  @RequestParameter
  public Account account;

  @Path("/{account}/entries/{entry}")
  public ActionResult index(){ … }
}
```

Converterで
自動変換！

その他Cubbyの特徴

- 設定ファイルレス
- Maven2による雛形作成
- 直感的なカスタムタグ
- プログラムで書くValidation
- 続きはドキュメントで！

デモ

簡単なWikiを
つくってみます。

ページの構成

GET /pages/{page}

POST /pages/{page}

1.プロジェクトの作成

```
mvn archetype:generate
  -DarchetypeCatalog=http://
  cubby.seasar.org
cd wiki
mvn eclipse:eclipse
```

2.各種設定

- WTP Server登録
- jdbc.dicon
- s2jdbc.dicon

3.作成するファイル

- wiki.action.PageAction
- wiki.converter.PageConverter
- wiki.entity.Page
- wiki.service.WikiService
- JSP
  - index.jsp
  - edit.jsp

今後のロードマップ

- 2008年2月 Ver1.0
  - 基本機能提供
- 2008年8月 Ver1.1
  - 実践的機能の充実
- 2009年　　Ver2.0
  - 他のコンテナ(Spring, Guice)対応
  - Archetypeの機能追加
    - プロジェクトの立ち上げをもっと便利に
    - Scaffold
  - Extenstion
    - 今風アプリ作成のサポート
    - Flashによる進捗バー付きのファイルアップロードダイアログなど

まとめ

- CubbyはRESTfulなWebアプリケーション開発を助けるシンプルなフレームワークです。
  - クールなWebサービスから業務アプリまで。
- まずは触ってみてください。
  - チュートリアルやサンプルから。
- 要望・フィードバックはMLまで。

ご清聴ありがとうございました。
ご質問があればどうぞ。

CM：
ヌーラボでは現在東京メンバーを募集中です。
詳しくはWebで！