FMCakeMix

Use FileMaker in an MVC web development framework.

ユーザーガイド

目次

イントロダクション

FMCakeMixは、MVCフレームワークであるCakePHP用のFileMakerデータソースドライバーです。FMCakeMixを利用すると、SQLデータベースと同じようにFileMakerとCakePHPを統合できます。すなわち、モダンなWebアプリケーションフレームワークを使って、FileMakerデータベースと連係するWebアプリケーションを迅速に開発できるようになるのです。

このガイドでは、CakePHPで利用できるFMCakeMixドライバーの利用方法の基本について、記述例を交えながら解説しています。

CakePHPに関する詳細については次のWebサイトを参照してください：http://cakephp.org/

インストール

CakePHP

http://cakephp.org/からCakePHPをダウンロードして、Webサイトのマニュアルにあるインストール手順にならってインストールおよび設定を行います。

FX.php

FX.phpは、Chris Hansen氏が中心になって開発した、PHPからFileMaker Proデータベースに接続するためのライブラリクラスです。FMCakeMixは、FileMaker Proデータベースに接続する際に内部的にFX.phpを利用しています。http://www.iviking.org/FX.php/からFX.phpのファイルをダウンロードして、FX.php、FX_Error.php、FX_Constants.phpおよびimage_proxy.phpのファイルをvendorsフォルダの直下に配置します。

FileMaker

FileMaker Proデータベースとの接続にXMLを利用しているため、XMLを使用したカスタムWeb公開機能をサポートしているFileMaker ServerもしくはFileMaker Server Advancedでデータベースをホストしなければなりません。手順についてはFileMaker Serverに付属のマニュアルを参照してください。

FMCakeMix

（CakePHP 1.3の場合には）app/models/datasources/dboフォルダにdbo_fmcakemix.phpファイルを配置します。おそらくdatasourcesフォルダにおいてdboという名称のディレクトリを作成する必要があるでしょう。

データベース接続の定義

データベース接続情報はapp/config/database.phpで定義します。FMCakeMixドライバーを使用して、以下のようにCakePHPからFileMakerへの接続に必要な詳細情報を記述します。複数のFileMaker Proデータベースファイルを参照する場合には、モデルを定義する際に設定情報を上書きできるので特に困ることはありません。

```
    var $default = array(
         'driver' => 'fmcakemix',
         'persistent' => false,
         'dataSourceType' => 'FMPro7',
         'scheme' => 'http',
         'port' => 80,
         'host' => '127.0.0.1',
         'login' => 'myUserName',
         'password' => 'myPassword',
         'database' => 'FMServer_Sample',
         'prefix' => '',
         'encoding' => 'utf8',
    );
```

モデルの定義

CakePHPの通常の流儀でモデルを定義できます。ただし、いくつかの機能は利用できない制限があるので、その詳細については「既知の制限事項」のセクションを参照するようにしてください。name、useDbConfigおよびprimaryKeyといったCakePHPで通常使用するモデルのプロパティに加えて、defaultLayoutプロパティでCakePHPのモデルとFileMakerのレイアウトを関連付けます。さらにfmDatabaseNameプロパティでFileMakerのデータベースファイル名を指定します。

hasMany、hasOne、belongsToおよびhasAndBelongsToManyプロパティを使って関連の形式を定義しますが、現状のドライバーではhasManyとbelongsToのみをサポートしています。FileMakerで関連データを処理する際、主に2つの方法があります。CakePHPで定義された関連の形式を使う方法、あるいはFileMakerが持つ機能であるポータルを通じてデータの関連付けや取得を行う方法です。CakePHPで定義された関連の形式を通じてデータを取得すると、すべての関連するモデル情

報を得るのにその都度データベースに接続することになるので、パフォーマンス上好ましくない影響があり得る点については注意するようにしてください。

```
<?php
class Book extends AppModel {

    var $name = 'Book';
    var $useDbConfig = 'default';
    var $primaryKey = 'ID';

    // FMCakeMix固有のプロパティ
    var $defaultLayout = 'web_books_general';
    var $fmDatabaseName = 'FMServer_Sample';

    // 関連モデルの定義（オプション）
    var $hasMany = array(
         'Comment' => array(
              'foreignKey' => '_fk_book_id'
         ),
         'History' => array(
              'foreignKey' => '_fk_book_id'
         )
    );

    // モデルで使用するバリデーションの条件を定義（オプション）
    var $validate = array(
         'Title' => array(
              'rule' => 'notEmpty'
         ),
         'Author' => array(
              'rule' => 'notEmpty'
         )
    );
}
```

コントローラーの記述例

コントローラーには入出力に関する処理を記述します。データベースの設定ファイルで接続に関する詳細を定義し、データベース処理はモデルで定義することで、コントローラーではアプリケー

ションのロジックやモデルをどう扱うかといったことに集中できます。FileMakerデータベースにおけるCRUD（データの生成、読み取り、更新、削除）の基本については以下で解説します。

生成（Create）

save

saveメソッドはデータを追加するときに使う基本的なメソッドです。CakePHPではフォームから入力したデータは$this->dataに格納されますが、FileMakerデータベースに新しいレコードを作成する際には2つのモデルメソッドを呼び出します。まず最初に新規レコードを保存するモデルを準備するためにcreateメソッドを、次にフォームで入力したデータを渡すためにsaveメソッドを利用します。CakePHPではある決まった名称のフィールドは特別扱いされるものがあり、例えば*created*や*modified*という名称のフィールドはタイムスタンプが自動的に保存されるようになっているという点にも留意する必要があります。

```
    function add() {
         if (!empty($this->data)) {
              $this->Book->create();
              if ($this->Book->save($this->data)) {
                   $this->Session->setFlash(__('The Book has been saved', true));
                   $this->redirect(array('action'=>'index'));
              } else {
                   $this->Session->setFlash(__('The Book could not be saved. Please,
try again.', true));
              }
         }
    }
```

saveAll

saveAllモデルメソッドを使うと複数のレコードを作成できます。 saveAllメソッドを使う場合には、CakePHPがトランザクション処理を使わないように常にatomicオプションにfalseを指定するようにします。

```
         $_data = array(
              'Comment' => array(
                   array(
                        '_fk_article_id' => $this->Book->getID(),
                        'body' => 'New Comment'
                   ),
                   array(
                        '_fk_article_id' => $this->Book->getID(),
```

```
                    'body' => 'Another Comment'
                )
            )
        );
        $this->loadModel('Comment');
        $this->Comment->create();
        $this->Comment->saveAll($_data['Comment'], array('atomic' => FALSE));
```

読み取り（Read）

find

下記の例は基本的な検索機能の実装例です。検索キーワードをタイトルに含み、かつ公開フラグの値が1になっているレシピのレコードを検索します。そして、*set*メソッドを使って結果セットのデータをビューに渡しています。

```
    function search() {
        $query = $this->data['Recipe']['title'];

        $recipes = $this->Recipe->find('all', array(
            'conditions' => array(
                'title' => $query,
                'published' => '=' . 1
            )
        ));

        $this->set('recipes', $recipes);
    }
```

paginate

下記はコントローラー内のindexメソッドで書籍の一覧をページ送りで表示する例です。ここでは、Bookモデルのrecursiveプロパティを0にして、不要な関連レコードを取得するリクエストが発生しないようにしています。

```
    var $paginate = array('limit' => 10, 'page' => 1);

    function index() {
        $this->Book->recursive = 0;
        $this->set('books', $this->paginate('Book'));
    }
```

削除（Delete）

del, remove

*del*メソッドおよびその別名の*remove*メソッドを使うと、データベースから1件のレコードを削除します。FileMakerでレコード削除処理を実行する際、削除したいレコードの内部レコードIDを指定することが必要となります。レコードID（-recid）の情報は、モデルのfindメソッドなどを利用した場合に返ってくるデータセットの中に含まれています。（訳注：下記のサンプルは動作しません。また、*del*メソッドおよび*remove*メソッドはCakePHP 1.3で利用できなくなりました。）

```
    function delete() {
         $this->Book->find('first', array(
              'conditions' => array(
                   'Book.ID' => 48
              ),
              'recursive' => 0
         ));

         $model->del();
    }
```

deleteAll

削除処理の実装例です。削除するレコードのレコードID（-recid）を渡し、実行結果を表示します。ここでは*del*メソッドの代わりに*deleteAll*メソッドを使って、任意のレコードを削除しています。

```
    function delete($recid = null) {
         if (!$recid) {
              $this->Session->setFlash(__('Invalid id for Book', true));
              $this->redirect(array('action'=>'index'));
         }
         if ($this->Book->deleteAll(array('-recid' => $recid), false)) {
              $this->Session->setFlash(__('Book deleted', true));
              $this->redirect(array('action'=>'index'));
         } else {
              $this->Session->setFlash(__('Book could not be deleted', true));
              $this->redirect(array('action'=>'index'));
         }
    }
```

更新（Update）

save

更新処理は生成処理と同じくモデルの*save*メソッドを使いますが、FileMakerの仕様上、レコードの編集にはレコードID（-recid）を指定する必要があります。下記の例では、-recidがフォームのhiddenで指定されているという前提で処理が記述されています。

```
    function edit($id = null) {
         if (!$id && empty($this->data)) {
              $this->Session->setFlash(__('Invalid Book', true));
         }
         if (!empty($this->data)) {
              if ($this->Book->save($this->data)) {
                   $this->Session->setFlash(__('The Book has been saved', true));
                   $this->redirect(array('action'=>'index'));
              } else {
                   $this->Session->setFlash(__('The Book could not be saved', true));
              }
         }
         if (empty($this->data)) {
              $this->data = $this->Book->read(null, $id);
         }
    }
```

ビューの記述例

フィールドと連動したフォーム

FMCakeMixはモデルを通じてデータベースに関する基本構造の情報を把握しているので、CakePHPのフォームヘルパーで入力フォームを作成すると、自動的にフォームとデータベースが連動するようになっています。

```
<?php echo $form->create('Book');?>
    <fieldset>
        <legend><?php __('Edit Book');?></legend>
    <?php
        echo $form->input('Title');
        echo $form->input('Author');
```

```
        echo $form->input('Publisher');
        echo $form->input('Status');
        echo $form->input('Description', array('type' => 'textarea'));
        echo $form->input('Quantity in Stock');
        echo $form->input('Number of Pages');
        echo $form->hidden('-recid');
    ?>
    </fieldset>
<?php echo $form->end('Submit');?>
```

ページ送り

コントローラーのサンプルでページ送りについて触れましたが、そこで使用したindexメソッド用のビューの実装例です。

```
<?php
echo $paginator->counter(array(
'format' => __('Page %page% of %pages%, showing %current% records out of %count%
total, starting on record %start%, ending on %end%', true)
));
?></p>
<table cellpadding="0" cellspacing="0">
<tr>
    <th><?php echo $paginator->sort('Title');?></th>
    <th><?php echo $paginator->sort('Author');?></th>
    <th><?php echo $paginator->sort('Publisher');?></th>
    <th class="actions"><?php __('Actions');?></th>
</tr>
<?php
$i = 0;
foreach ($books as $book):
    $class = null;
    if ($i++ % 2 == 0) {
        $class = ' class="altrow"';
    }
?>
    <tr<?php echo $class;?>>
        <td>
            <?php echo $html->link($book['Book']['Title'], array
('controller'=>'books', 'action'=>'view', $book['Book']['ID'])); ?>
        </td>
        <td>
```

```
            <?php echo $book['Book']['Author']; ?>
        </td>
        <td>
            <?php echo $book['Book']['Publisher']; ?>
        </td>
        <td class="actions">
            <?php echo $html->link(__('View', true), array('action'=>'view', $book
['Book']['ID'])); ?>
            <?php echo $html->link(__('Edit', true), array('action'=>'edit', $book
['Book']['ID'])); ?>
            <?php echo $html->link(__('Delete', true), array('action'=>'delete', $book
['Book']['-recid'], null, sprintf(__('Are you sure you want to delete $s?', true),
$book['Book']['Title'])); ?>
        </td>
    </tr>
<?php endforeach; ?>
</table>
<div class="paging">
    <?php echo $paginator->prev('<< '.__('previous', true), array(), null, array
('class'=>'disabled'));?>
    <?php echo $paginator->numbers();?>
    <?php echo $paginator->next(__('next', true).' >>', array(), null, array
('class'=>'disabled'));?>
</div>
```

既知の制限事項

FileMaker

- オブジェクトフィールド：オブジェクトフィールドにファイルをアップロードすることはできません。

CakePHPのモデル

関連の形式

- hasOne：現状ではこの関連の形式はサポートされていません。
- hasAndBelongsToMany： 現状ではこの関連の形式はサポートされていません。

メソッド

- deleteAll：-recidが一致するレコードしか削除できないため、一度に複数のレコードを削除する処理はサポートされていません。また、関連レコードが削除されないように、このメソッドの第2引数にfalseを指定する必要があります。

- save：フィールドリストのパラメーターや、保存対象とするフィールドのホワイトリストはサポートされていません。

- saveAll：トランザクションをサポートしていないため、atomicオプションは必ずfalseが指定されていなければなりません。