GoodfindEngineerACADEMY

フレームワークによるWebアプリ開発
管理画面編

Goodfind Engineer Academy
<3 Days “Programming Bootcamp”>

無線LANにご接続ください

SSID ： slogan-guest (- 5g)　　PASS ： welcomeslogan

このセミナーの目的

・Webページからのデータの操作を一通り理解する

・MVCフレームワークでの効率的な開発の流れを理解する

・「なんとなく作れそう！」という自信を得る

Agenda

本日のアジェンダ

1. 更新系画面の開発

GETとPOST(データ送信の形式)

- データをURLに含めて送信するのが「GET」
- データをURLに含まずに送信するのが「POST」

POSTでデータを送信する場合は、formタグを使う

```
<form action=“/users/login” method=“post”>

</form>
```

POSTで送信するデータの入力は、inputタグを使う

```
<form action=“/users/login” method=“post”>

  <input name=“id”>
  <input name=“title”>
  <input name=“body”>

</form>
```

データの入力形式の指定には、

type属性を使う(textarea, hiddenなど)

```
<form action="/users/login" method="post">

  <input name="id" type="hidden">
  <input name="title">
  <input name="body" type="textarea">

</form>
```

送信ボタンは
type属性にsubmitを指定する

```
<form action="/users/login" method="post">

  <input name="id" type="hidden">
  <input name="title">
  <input name="body" type="textarea">

  <input value="send" type="submit">
</form>
```

CakePHPでは、formの生成には

FormHelperを使う！

(View上で、$this->Form で呼び出せる)

formタグは create 〜 end で生成

(createには対応するModel名を入力)

```
<?php echo $this->Form->create('Topic'); ?>

<?php echo $this->Form->end(); ?>
```

データの入力項目には、
input関数を使う

```
<?php echo $this->Form->create('Topic'); ?>
  <?php echo $this->Form->input(); ?>
<?php echo $this->Form->end(); ?>
```

inputには、対応するカラム名を渡す

(name属性に対応)

```
<?php echo $this->Form->create('Topic'); ?>
  <?php echo $this->Form->input('body'); ?>
<?php echo $this->Form->end(); ?>
```

データの入力形式の指定には、

2番目の引数でtypeを設定(text, textarea, hiddenなど)

```
<?php echo $this->Form->create('Topic'); ?>
  <?php echo $this->Form->input('id',
        array('type' => 'hidden')); ?>
  <?php echo $this->Form->input('body',
        array('type' => 'textarea')); ?>
<?php echo $this->Form->end(); ?>
```

送信ボタンは
submit関数を利用する

```
<?php echo $this->Form->create('Topic'); ?>
  <?php echo $this->Form->input('id',
        array('type' => 'hidden')); ?>
  <?php echo $this->Form->input('body',
        array('type' => 'textarea')); ?>
  <?php echo $this->Form->submit('登録'); ?>
<?php echo $this->Form->end(); ?>
```

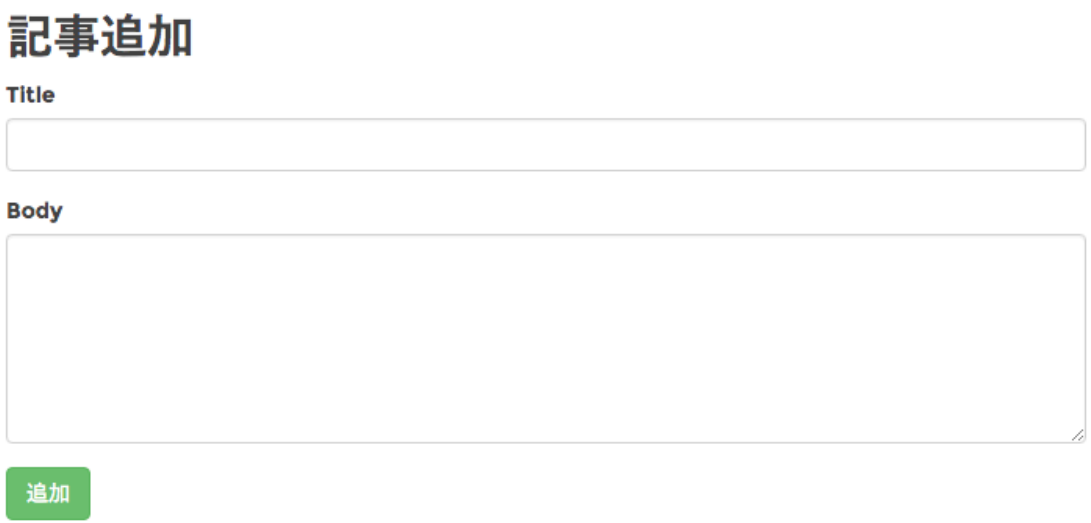
記事を追加する画面を作成してみよう！

- URLは、
  [トップページのURL]/topics/add

- formの中にタイトルと本文の入力欄をつくる

- デザインもできれば整えてみよう！

このような画面を
作成してみよう！



app/View/Topics/add.ctp

```
<div class="row">
  <div class="col-lg-offset-3 col-lg-6 col-md-offset-2 col-md-8 col-xs-offset-1 col-xs-10">
    <h2>記事追加</h2>
    <?php echo $this->Form->create('Topic', array('class' => 'form')); ?>
    <div class="form-group">
      <?php echo $this->Form->input('title', array('class' => 'form-control')); ?>
    </div>
    <div class="form-group">
      <?php echo $this->Form->input('body', array('type' => 'textarea', 'class' => 'form-control')); ?>
    </div>
    <div class="form-group">
      <?php echo $this->Form->submit('追加', array('class' => 'btn btn-success')); ?>
    </div>
    <?php echo $this->Form->end(); ?>
  </div>
</div>
```

POSTされたデータは、Controllerの$this->request->data に渡される

```
var_dump($this->request->data);

↓

array(
  'Topic' => array(
    'title' => '吾輩は猫である',
    'body' => '吾輩は猫である。名前はまだない。…(略)'
  )
)
```

データの送信形式がPOSTかどうかは $this->request->is() 関数で判定する

```
if ($this->request->is('post')) {

    (保存するときの処理)

} else {

    (表示するときの処理)

}
```

データを保存する

Model::save()

を使う

Model::save()　＝　INSERT or UPDATE

(どちらになるかは、’id’の値があるかで決まる)

引数には、Modelと同名のキーを含む配列を渡す

Model::save()実行前に、Model::create()が必要

```
$this->Topic->create();

$this->Topic->save($this->request->data);
```

記事を追加する処理(Controller)を作成してみよう！

- URLは、
  [トップページのURL]/topics/add

- POSTで送信された場合にのみ、保存処理を実行

- 入力項目のないカラムには仮の値(1など) を入れる

まずは、POST形式かどうかを判定

app/Controller/TopicsController.php

```
public function add() {
    if ($this->request->is('post')) {

    }
}
```

save() 関数で入力したデータを保存

```
public function add() {
    if ($this->request->is('post')) {
        $this->Topic->create();
        $this->Topic->save($this->request->data);
    }
}
```

category_id, user_id に仮の値を入力

```
public function add() {
    if ($this->request->is('post')) {
        $this->request->data['Topic']['category_id'] = 1;
        $this->request->data['Topic']['user_id'] = 1;

        $this->Topic->create();
        $this->Topic->save($this->request->data);
    }
}
```

カテゴリを選択入力できるようにする

カテゴリー覧を取得してViewにset

```
public function add() {
    if ($this->request->is('post')) {
        //$this->request->data['Topic']['category_id'] = 1;
        $this->request->data['Topic']['user_id'] = 1;

        $this->Topic->create();
        $this->Topic->save($this->request->data);
    }
    $categories = $this->Category->find('list');
    $this->set('categories', $categories);
}
```

カテゴリの選択欄をつくる

app/View/Topics/add.ctp

```
    (略)
    <div class="form-group">
      <?php echo $this->Form->input('body', array('type' => 'textarea', 'class' => 'form-control')); ?>
    </div>
    <div class="form-group">
      <?php echo $this->Form->input('category_id', array('type' => 'select', 'class' => 'form-
control')); ?>
    </div>
    <div class="form-group">
      <?php echo $this->Form->submit('追加', array('class' => 'btn btn-success')); ?>
    </div>
    <?php echo $this->Form->end(); ?>
  </div>
</div>
```

Model::save()の返り値は

成功したら true

失敗したら false

保存が成功した場合はトップページにもどる

app/Controller/TopicsController.php

```
public function add() {
    if ($this->request->is('post')) {
        //$this->request->data['Topic']['category_id'] = 1;
        $this->request->data['Topic']['user_id'] = 1;

        $this->Topic->create();
        if ($this->Topic->save($data)) {
            $this->redirect('/');
        }
}
    $categories = $this->Category->find('list');
    $this->set('categories', $categories);
}
```

[演習]記事詳細にコメント追加機能を作成してみよう！

- URLは、
  [トップページのURL]/detail/[:id]

- 名前・タイトル・コメントの入力欄をつける

- POST判定、save()関数の使い方を確認しながら…

データを削除する

Model::delete()

を使う

Model::delete()　＝　DELETE

引数には、Modelのプライマリキーの値を渡す
(基本的にはidの値)

```
$this->Topic->delete($id);
```

Model::delete()の返り値も

成功したら true

失敗したら false

記事の削除を完成させよう！

- URLは、
  [トップページのURL]/topics/delete/[:id]

- 記事一覧(トップページ)に、「削除」ボタンを設置

- 削除後は、トップページに戻る

削除のボタンを
記事一覧に設置

削除ボタンの実装例

app/View/Pages/index.ctp

```
<button type="button" class="btn btn-default">
    <?php echo $this->Html->link(
        'Delete',
        array(
            'controller' => 'topics',
            'action' => 'delete',
            $topic['Topic']['id']
        )
    ); ?>
</button>
```

delete.ctp を作る必要はない！

これだけでもOK (idが正常値なら)

app/Controller/TopicsController.php

```
class TopicsController extends AppController
{
    (略)
    public function delete($id = null) {
        $this->Topic->delete($id);
        $this->redirect('/');
    }
}
```

Model::delete() の実行結果により メッセージを分ける

```
class TopicsController extends AppController
{
    public $components = array('Session');
    (略)
    public function delete($id = null) {
        if ($this->Topic->delete($id)) {
            $this->Session->setFlash('成功しました');
        } else {
            $this->Session->setFlash('失敗しました');
        }
        $this->redirect('/');
    }
}
```

トップページで実行結果のメッセージを表示

app/View/Pages/index.ctp

```
(略)
    <div class="alert alert-success">
        <?php echo $this->Session->flash(); ?>
    </div>
    <?php foreach ($topics as $topic): ?>

(略)
```

[演習]記事の編集画面を完成させよう！

- URLは、
  [トップページのURL]/topics/edit/[:id]

- POSTでない場合はModel::findを使ってSELECT
  POSTの場合はModel::saveを使ってUPDATE

- hiddenタグなどを使ってidをPOST送信しないと、
  Model::saveではINSERT扱いになるので注意！

Model::find()　＝　SELECT

Model::save()　＝　INSERT or UPDATE

Model::delete()　＝　DELETE

データの操作がすべて揃った！