Javascript et codeIgniter

< Introduction à CodeIgniter >>

1. Introduction à CodeIgniter
2. Helpers
3. Bibliothèques
4. Sessions CodeIgniter
5. Validation des formulaires
6. ORM IgnitedRecord
7. Doctrine
8. Javascript et codeIgniter

Solutions possibles

Il est possible d'utiliser les frameworks javascript connus :

- JQuery
- Prototype
- Mootools
- Ext Js

ou de mettre en oeuvre un framework Javascript s'intégrant à CodeIgniter : CJAX

CJAX Framework

- Documentation doc sur sf Doc sur Google code
- Téléchargement

Installation

Copier le contenu du zip dans les destinations suivantes :

- **ajax.php** et **ajaxfw.php** et le dossier **cjax** dans **/**
- le contenu du dossier **application** dans le dossier **application** du site
  - le dossier **response**
  - le contenu de **views** vers **views**
  - le contenu de **controllers** vers **controllers**

Configuration

Dans **cjax/config.php**, mettre le fallBack à true :

```
$config->fallback = true;
```

Mise en oeuvre

Avec CJAX, Les contrôleurs se situent dans le dossier **response** de application, et sont accessibles par l'url **ajax.php?nomController**.

Message sur click

Afficher un message sur la page sur le click d'un élément :

- Créer un contrôleur dans **response** nommé **TestAjax**

|h application/response/testAjax.php

```
<?php
class TestAjax extends CI_Controller{
    public function click(){
        $this->load->view('vClick');
    }
}
?>
```

- Créer la vue **vClick.php** dans **applications/views/**
  - **ajax()** permet d'instancier ajax ou de récupérer l'instance active
  - **$ajax->init()** insère le fichier javascript CJAX dans le header de la page
  - **$ajax->exec** associe l'appel de **update** au click sur le bouton **btClick**
  - **$ajax->update** affiche le message “Test de click” dans **divResponse**

application/views/vClick.php

```
<?php
require_once "ajax.php";
$ajax = ajax();
$ajax->Exec("btClick",$ajax->update("divResponse","Test de click"));

?>
<html>
<head>
<meta charset="UTF-8">
<title>Message sur click</title>
<?php echo $ajax->init();?>
</head>
<body>
<h2>message sur click du button</h2>
<input type='button' id='btClick' value='Cliquer sur le bouton'>
<div id="divResponse">divResponse</div>
</body>
</html>
```

Requête sur click

Il s'agit de faire une requête vers une page (/test/affiche/) sur le click d'un bouton, et d'afficher le résultat dans une div

- Ajouter une méthode dans le controleur **testAjax** :

|h application/response/testAjax.php

```
<?php
class TestAjax extends CI_Controller{
    public function click(){
        $this->load->view('vClick');
    }
    public function get(){
        $this->load->view('vGet');
    }
}
?>
```

- Créer la vue **vGet.php**
  - **$ajax->exec** exécute l'appel du **call** sur le click du bouton **btGet**
  - **$ajax->divResponse** permet d'affecter le résultat du chargement de la page **affiche** à la div **divResponse**

application/views/vGet.php

```
<?php

require_once "ajax.php";
$ajax = ajax();

$ajax->Exec('btGet' , $ajax->divResponse=$ajax->call("test/affiche"));

?>
<html>
<head>
<meta charset="UTF-8">
<title>GET</title>
<?php echo $ajax->init();?>
</head>
<body>
<h2>Get</h2>
<input type='button' id='btGet' value='Cliquer sur le bouton'>
<div id="divResponse">divResponse</div>
</body>
</html>
```

POST sur click

Poster des variables vers une URL sur le click d'un élément :

- Ajouter 2 méthodes dans le contrôleur testAjax :
  - **post** permet d'afficher la vue permettant de poster une requête
  - **postResult** affiche le résultat du POST

|h application/response/testAjax.php

```
<?php
class TestAjax extends CI_Controller{
    public function click(){
        $this->load->view('vClick');
    }
    public function get(){
        $this->load->view('vGet');
    }
    public function post(){
        $this->load->view('vPost');
    }
    public function postResult(){
        var_dump($_POST);
    }
}
?>
```

- Créer la vue **vPost.php**
  - **$ajax->post** définit les variables postées

application/views/vPost.php

```
<?php
require_once "ajax.php";
$ajax = ajax();
$vars = array(
        'hello' => 'world',
        'world' => 'hello!',
        'someVar' => 'someValue',
        'x' => 'y'
);

$ajax->post = $vars;
$ajax->Exec('btPost' ,
$ajax->call("ajax.php?testAjax/postResult",'divResponse'));

?>
<html>
<head>
<meta charset="UTF-8">
<title>POST</title>
<?php echo $ajax->init();?>
</head>
<body>
<h2>POST</h2>
<input type='button' id='btPost' value='Cliquer sur le bouton'>
<div id="divResponse">divResponse</div>
</body>
```

```
</html>
```

Envoi de formulaire

Soumettre un formulaire existant vers une URL :

- Ajouter 2 méthodes au contrôleur :
  - **getForm** affiche la vue vForm (formulaire)
  - **formSubmit** récupère le résultat du submit de vForm

|h application/response/testAjax.php

```
    public function getForm(){
        $this->load->view('vForm');
    }
    public function formSubmit(){
        $ajax = ajax();
        $ajax->alert("Champs soumis : ".print_r($_POST,1));
    }
```

- Créer la vue **vForm.php**
  - **$ajax->form** soumet le formulaire à **testAjax/formSubmit**

application/views/vForm.php

```
<?php
require_once "ajax.php";
$ajax = ajax();
$ajax->click("btSubmit",$ajax->form("ajax.php?testAjax/formSubmit","frm"));

?>
<html>
<head>
<meta charset="UTF-8">
<title>SubmitForm</title>
<?php echo $ajax->init();?>
</head>
<body>
<h2>SubmitForm</h2>
<form id="frm" name="frm" method="post" action="">
    <div><label for="nom">Nom : </label><input type="text" id="nom"
name="nom"></div>
    <div><label for="prenom">Prénom : </label><input type="text" id="prenom"
name="prenom"></div>
    <input type='submit' id='btSubmit' value='Valider'>
</form>
<div id="divResponse"></div>
</body>
</html>
```

Envoi vers une zone HTML

- Modifier la méthode du contrôleur en :

```
public function formSubmit(){
    $ajax = ajax();
    $ajax->divResponse="Champs soumis : ".print_r($_POST,1);
}
```

Implémentez les fonctionnalités précédentes en AJAX :

- Pour les utilisateurs :
  - Afficher la liste des utilisateurs, et leur catégorie
  - Créer la fonctionnalité d'ajout d'un utilisateur
  - Créer la fonctionnalité de modification d'un utilisateur existant
  - Créer la fonctionnalité de suppression d'un utilisateur
- Pour les catégories :
  - Afficher la liste des catégories, et les utilisateurs correspondants
  - Créer la fonctionnalité d'ajout de catégorie
  - Créer la fonctionnalité de modification d'une catégorie existante
  - Créer la fonctionnalité de suppression d'une catégorie

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/slam4/php/codeigniter/javascript?rev=1355994557

Last update: **2019/08/31 14:41**