Uma forma de fazer requisições Ajax juntamente com o framework CodeIgniter.

Posted on 28/05/2015 by Cristian da Silva in AJAX, Framework, HTML, Java Script, jquery, Orientação a Objetos, PHP, Programação

Uma forma de fazer requisições Ajax juntamente com o framework CodeIgniter.

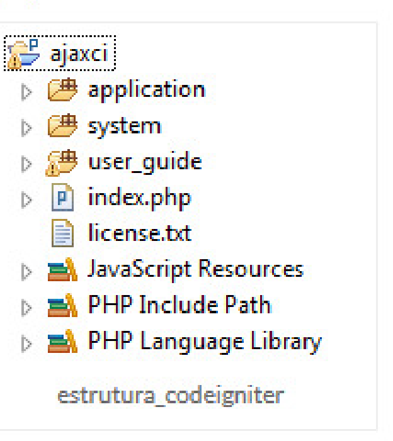
Olá como vão vocês? Nesse post será mostrado uma forma de fazer requisição **Ajax,** juntamente com o framework **CodeIgniter**. O que vai ser feito? Resumidamente será gravada informação para o banco de dados sem atualizar o formulário.

Lembrando que essa é uma das formas, existem vários outras formas de enviar os dados para o banco de dados, sem que a página seja recarregada. Então vamos ao trabalho.

Antes de tudo, instale um servidor em sua máquina, neste exemplo irá ser utilizado o **xampp**, pode-se utilizar qualquer um dos servidores existente na internet. Se não souber instalar o **xampp** veja esse link que ajudará a instala-lo.

E agora? Bom depois de instalado o servidor, baixe o **CodeIgniter** e crie um projeto novo com o nome de **ajaxci**, se ainda não sabe como fazer este processo, veja o artigo **Sistema de login usando Framework PHP Codeigniter,** que mostra como fazer a instalação do **CodeIgniter.**

Depois de pronto e instalado, a estrutura do projeto ficará dessa forma...



```
ajaxci
  application
  system
  user_guide
  index.php
  license.txt
  JavaScript Resources
  PHP Include Path
  PHP Language Library
```

estrutura_codeigniter

"A IDE de programação usada vai ser o **eclipse,** mas, nada impede de ser usada outra IDE de programação como Sublime Text, Notepad++ ou Netbeans."

Pronto... Abra a pasta **application,** em seguida encontre o arquivo **autoload.php** e coloque as seguintes configurações no arquivo.

```
$autoload['libraries'] = array('database');
$autoload['helper'] = array('url', 'form');
```

Essas linhas de códigos indicam que usaremos para carregar automaticamente as bibliotecas que irão auxiliar na comunicação ao banco de dados, e carregará alguns **helpers(ajudas)** para o projeto.

Chega a hora de configurar o banco de dados na nossa aplicação, continue na pasta **application,** e localize o arquivo **database.php**, e modifique as seguinte linhas de códigos:

```
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = '';
$db['default']['database'] = 'clientes';
$db['default']['dbdriver'] = 'mysql';
$db['default']['dbprefix'] = '';
$db['default']['pconnect'] = TRUE;
$db['default']['db_debug'] = TRUE;
$db['default']['cache_on'] = FALSE;
$db['default']['cachedir'] = '';
$db['default']['char_set'] = 'utf8';
$db['default']['dbcollat'] = 'utf8_general_ci';
$db['default']['swap_pre'] = '';
$db['default']['autoinit'] = TRUE;
$db['default']['stricton'] = FALSE;
```

configuração database

Neste exemplo será usado o banco de dados **MySQL**, que vem por padrão no **xampp**, mas pode ser utilizado um de sua preferencia, somente use a documentação do **CodeIgniter** para lhe auxiliar na configuração.

Link para a documentação do CodeIgniter.

Explicando as principais linhas do código acima...

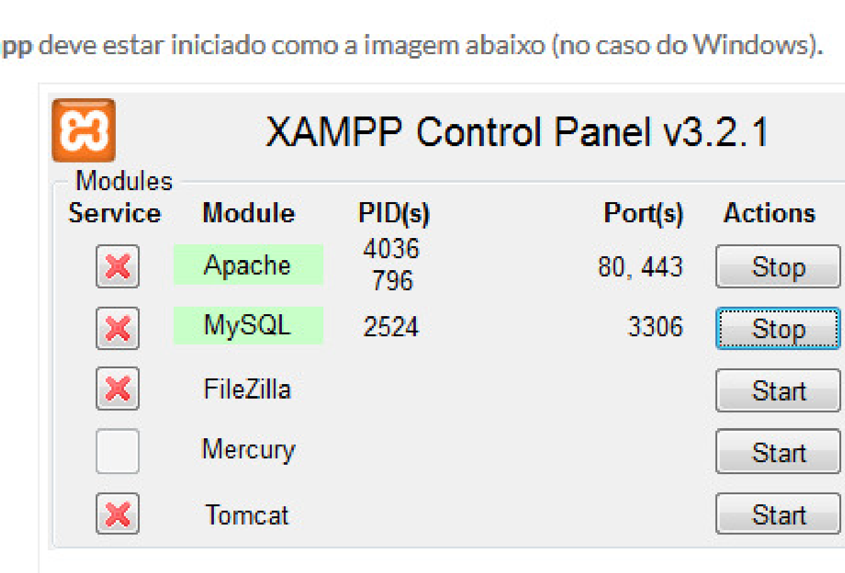
Na primeira linha, será configurado o servidor onde está o banco de dados, segunda linha configura-se o usuário do banco de dados, neste caso root. Quarta linha, senha do banco de dados (por padrão do **xampp** vem sem senha), configurando a quinta linha, vem o nome do banco de dados, (neste exemplo vamos fazer um cadastro de usuários) e sexta linha é a configuração do driver que vai ser utilizado, neste caso o **MySQL.**

Criação do banco de dados.

Quase pronto. O que precisa ser criado agora é o banco de dados com a sua tabela de clientes.

Veja se o seu servidor está iniciado, e digite em seu navegador preferido o seguinte endereço: **http://localhost/phpmyadmin.**

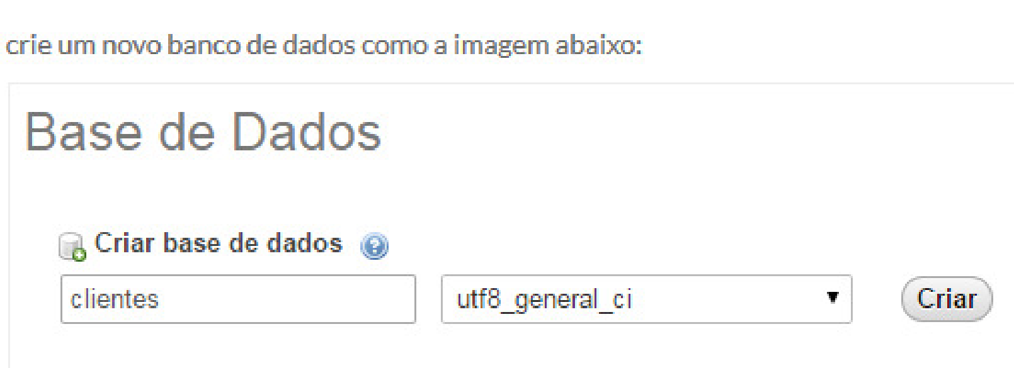
O gerenciador do **xampp** deve estar iniciado como a imagem abaixo (no caso do Windows).



Service	Module	PID(s)	Port(s)	Actions
☒	Apache	4036 796	80, 443	Stop
☒	MySQL	2524	3306	Stop
☒	FileZilla			Start
☐	Mercury			Start
☒	Tomcat			Start

xampp_iniciado

Clique em **new** e crie um novo banco de dados como a imagem abaixo:

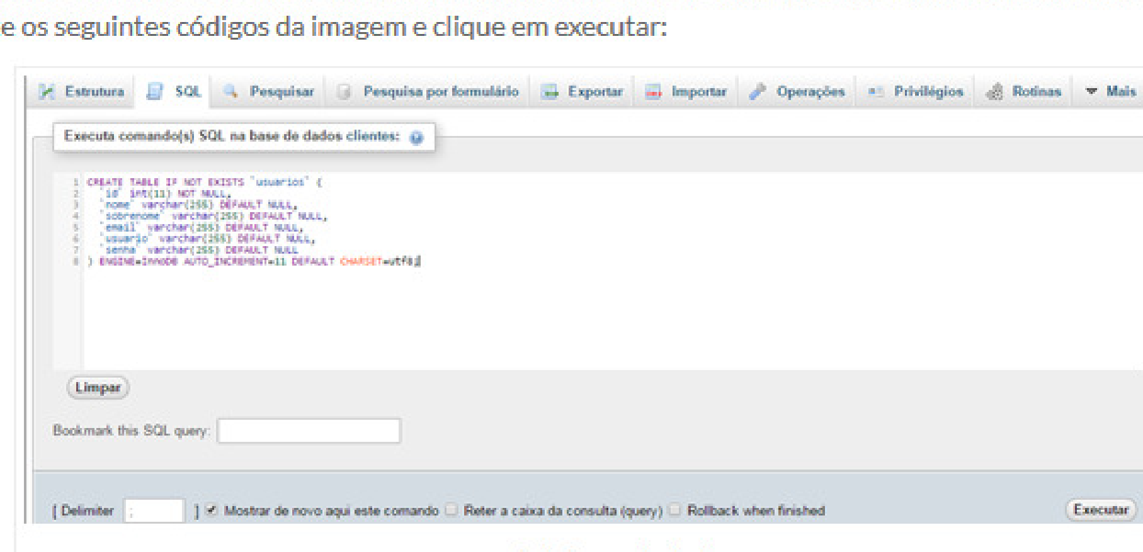


clientes

utf8_general_ci

tela criação novo banco de dados

Neste banco de dados vai existir uma tabela chamada **usuarios**, clique no banco que acabou de ser criado, na aba SQL e digite os seguintes códigos da imagem e clique em executar:



Executa comando(s) SQL na base de dados clientes:

```
CREATE TABLE IF NOT EXISTS `usuarios` (
  `id` int(11) NOT NULL,
  `nome` varchar(255) DEFAULT NULL,
  `sobrenome` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `usuario` varchar(255) DEFAULT NULL,
  `senha` varchar(255) DEFAULT NULL
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8;
```

[Delimiter ;] ☑ Mostrar de novo aqui este comando ☐ Reter a caixa da consulta (query) ☐ Rollback when finished

SQL tabela banco de dados

Pronto, criado o banco de dados com a tabela que gostaríamos, agora vamos para a parte da programação do sistema de cadastros de usuários.

Criando o projeto e importando o JavaScript no projeto.

Depois de seu projeto criado, vá à pasta **view**, crie um arquivo chamado cadastro, coloque o seguinte código nele:

```

<!doctype html>
<html>
<head>
<title>Requisições Ajax com CodeIgniter</title>

<script type="text/javascript"
    src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js">
</script>

<script type="text/javascript">

</script>
</head>

<body>

    <div id="mensagem"></div>
    <form id="formulario_cadastro">

    <?php echo form_label('Nome', 'nome');?>
    <?php echo form_input(array(
                "name" => "nome",
                "id" => "nome",
                "type" => "text",
                "placeholder" => "Nome"
                ));?>

                <?php echo form_label('Sobrenome', 'nome');?>
                <?php echo form_input(array(
                "name" => "sobrenome",
                "id" => "sobrenome",
                "type" => "text",
                "placeholder" => "Sobrenome"
                ));?>

                <?php echo form_label('Usuario', 'usuario');?>
                <?php echo form_input(array(
                "name" => "usuario",
                "id" => "usuario",
                "type" => "text",
                "placeholder" => "Usuario"
                ));?>

                <?php echo form_label('Senha', 'senha');?>
                <?php echo form_password(array(
                "name" => "senha",
                "id" => "senha",
                "type" => "text",
                "placeholder" => "Senha"
                ));?>

                <?php echo form_input(array(
                "name" => "cadastrar",
                "id" => "cadastrar",
                "type" => "submit"
                ), "Cadastrar");?>

    </form>

</body>
</html>
```

Na primeira tag **<script></script>** é feito a importação do **Jquery**, que auxilia nas requisições **Ajax.** Dentro da tag **<body></body>** está o nosso formulário, lembra o **helper 'form'** que foi configurado no arquivo **autoload.php**, então, este arquivos contém funções que auxiliam no trabalho com formulários.

Depois de criado esse arquivo, abra a pasta **controller**, lá vai ser criado o controlador da aplicação, o arquivo chamado **registros_controller.php** que terá os seguintes códigos:

```

<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Registros_Controller extends CI_Controller {
    public function index(){
        $this->load->view('cadastro');
    }

    public function cadatrar(){
        $this->load->model('registros_model');
        $this->registros_model->cadastrar();
        echo "Cadastrado com sucesso!!";
    }
}
```

Nesse **controller** cria-se uma função **index** para carregar a **view** principal, e uma segunda função para carregar o modelo e sua função para manipulação do bando de dados, em seguida mostra a mensagem para o usuário com o **echo.**

Agora cria-se o modelo, para que funcione a nossa função do **controller.**

Vá à pasta **model** e crie o arquivo chamado **registros_model.php** e coloque o seguinte código:

```

<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Registros_Model extends CI_Model {

    public function cadastrar(){

        // Inserção dos dados
        $this->db->insert('usuarios', array(

            'nome' => $this->input->post('nome', TRUE),
            'sobrenome' => $this->input->post('sobrenome', TRUE),
            'senha' => md5($this->input->post('senha', TRUE)),
            'usuario' => $this->input->post('usuario', TRUE)

        ));
    }
}
```

Bem, este arquivo só fará a inserção dos dados. Mas de onde ele pega os dados?

Fácil, com o método $this->input->post('id_campo', **TRUE**) ele irá pegar os dados que vier do formulário que tenho o **id** igual **nome, sobrenome, senha e usuario.**

Vamos para a "mágica".

Depois de feito o modelo, visão e o controlador **(model, view, controller** – padrão MVC), chegado ao assunto do post, a requisição Ajax.

Na segunda tag **<script></script>**, vai ser colocado o seguinte código:

```

<script type="text/javascript">
            $(document).ready(function(){
                $("#cadastrar").click(function(){
                        //alert("ola");
                        $.ajax({
                                url: '<?= base_url(); ?>' + 'index.php/registros_co
                                type: 'POST',
                                data: $("#formulario_cadastro").serialize(),
                                success: function(msg){
                                    $("#mensagem").html(msg);
                                    if(msg == 'Cadastrado com sucesso!!'){
                                        jQuery.fn.reset = function(){
                                                $(this).each(function(){ this.reset
                                            }
                                        $("#formulario_cadastro").reset();
                                    }
                                    }
                            });
                        return false;
                    });
            });
        </script>
```

Mas o que esse código exatamente faz?

Explicando então... Na primeira linha do código verifica quando o documento estiver carregado, execute uma determinada função anônima, após isso verifica que quando o botão com o **id 'cadastrar'** for clicado, execute outra função anônima, que consiste com os seguintes comandos:

Executará uma requisição **Ajax** com os seguintes parâmetros:

```
url: '<?= base_url(); ?>' + 'index.php/registros_controller/cadatrar'
```

O valor é o URL para a qual a requisição é feita.

```
type: 'POST'
```

Define o tipo de requisição, por padrão é **GET**, mas vamos colocar como post para encapsular as informações.

```
data: $("#formulario_cadastro").serialize(),
```

O valor dessa chave serão os dados a serem enviados ao servidor, e usará o método **serialize()** para coletar os valores entrados pelo usuário em campos do formulário e transforma-los em formato de string.

```

success: function(msg){
                                    $("#mensagem").html(msg);
                                    if(msg == 'Cadastrado com sucesso!!'){
                                        jQuery.fn.reset = function(){
                                                $(this).each(function(){ this.reset(
                                            }
                                        $("#formulario_cadastro").reset();
```

O valor é uma função que é executada quando a requisição se completa com sucesso. Depois de verificado executa a função que vai atualizar somente a **tag** que tiver o **id 'mensagem'** que neste caso é uma tag **<div></div>**, depois verifica se a mensagem que é retornada pelo método do **controller** é igual a **'Cadastro com sucesso!!'**, se for verdadeiro executa uma função para limpar todos os campos do formulário.

O arquivo inteiro ficará dessa forma:

```

<!doctype html>
<html>
<head>
<title>Requisições Ajax com CodeIgniter</title>
<script type="text/javascript"
    src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script

<script type="text/javascript">
            $(document).ready(function(){
                $("#cadastrar").click(function(){
                        //alert("ola");
                        $.ajax({
                                url: '<?= base_url(); ?>' + 'index.php/registros_co
                                type: 'POST',
                                data: $("#formulario_cadastro").serialize(),
                                success: function(msg){
                                    $("#mensagem").html(msg);
                                    if(msg == 'Cadastrado com sucesso!!'){
                                        jQuery.fn.reset = function(){
                                                $(this).each(function(){ this.reset
                                            }
                                        $("#formulario_cadastro").reset();
                                    }
                                    }
                            });
                        return false;
                    });
            });
        </script>
</head>

<body>

    <div id="mensagem"></div>
    <form id="formulario_cadastro">

    <?php echo form_label('Nome', 'nome');?>
    <?php echo form_input(array(
                "name" => "nome",
                "id" => "nome",
                "type" => "text",
                "placeholder" => "Nome"
                ));?>

                <?php echo form_label('Sobrenome', 'nome');?>
                <?php echo form_input(array(
                "name" => "sobrenome",
                "id" => "sobrenome",
                "type" => "text",
                "placeholder" => "Sobrenome"
                ));?>

                <?php echo form_label('Usuario', 'usuario');?>
                <?php echo form_input(array(
                "name" => "usuario",
                "id" => "usuario",
                "type" => "text",
                "placeholder" => "Usuario"
                ));?>

                <?php echo form_label('Senha', 'senha');?>
                <?php echo form_password(array(
                "name" => "senha",
                "id" => "senha",
                "type" => "text",
                "placeholder" => "Senha"
                ));?>

                <?php echo form_input(array(
                "name" => "cadastrar",
                "id" => "cadastrar",
                "type" => "submit"
                ), "Cadastrar");?>

    </form>

</body>
</html>
```

Depois de tudo terminado, falta somente testar, para facilitar as coisas, abra o arquivo **routes.php**, mude a seguinte linha de código para o nome do **controller** principal:

Mude este código:

```

$route['default_controller'] = "welcome";
$route['404_override'] = '';
```

Para este código:

```

$route['default_controller'] = "registros_controller";
$route['404_override'] = '';
```

Em fim, vá ao navegador de internet preferido internet Explorer, por exemplo, (rsrsrs), e digite o seguinte endereço, http://localhost/ajaxci/ e teste a sua aplicação e veja o resultado.

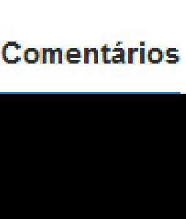
O resultado deverá ser o seguinte... Quando digitar os dados do usuário que irá cadastra, quando for clicado o botão cadastrar, irá mandar os dados digitados para o banco de dados, após ocorrer tudo como esperado mostra a mensagem de "cadastrado com sucesso!!" para o usuário, e em seguida limpar todos os campos do formulário.

Se caso não acontecer isso, verifique o seu código, e veja se está tudo correto como esta explicado no post.

Gostou do artigo? Ficou com dúvidas? Tem outras sugestões?
Deixe um comentário.
Forte Abraço e até a próxima!

24
FLARES

Tagged AJAX, artigo, classe, codeigniter, html, Orientação a objetos, php, programação, requisições, tutorial



Cristian da Silva

Bacharel em Ciência da computação, trabalhando originalmente na SoulClinic como startup e 9bits como desenvolvedor web em São Miguel do Oeste - SC. Certificados de cursos: HTML Básico, Java Básico, Java Intermediário, Java Desenvolvimento de Jogos Básico, Curso de Android, Curso de Orientação a Objetos com Java.

PREVIOUS
Sistema de login usando Framework PHP Codeigniter

NEXT
Palácio da memória, palácio mental ou memoria mnemônica.

2 Comentários www.ideiasprogramadas.com.br João Felipe

SIGA-ME

APLICAÇÕES

CATEGORIAS

- AJAX
- Android
- Angular
- AngulasJs
- Fortran
- Framework
- Games
- Hello Word
- HTML
- JAVA
- Java Script
- jquery
- Oficinas
- PHP
- Programação
- Python

BLOGROLL

TAGS