

Implantación de Sistemas

Guía 4 –Acceso a Datos con CodeIgniter

Instructor: Ing. David Gildardo Rajo





Contenido

| | |
|---|----|
| ACCESO A DATOS CON CODEIGNITER | 3 |
| CONFIGURANDO LA CONEXIÓN A LA BASE DE DATOS. | 3 |
| CONFIGURACION DEL MODELO | 4 |
| FUNCIONES HELPER | 6 |
| Creando el controlador. | 7 |
| CREANDO LAS VISTAS..... | 8 |
| EJERCICIOS..... | 10 |

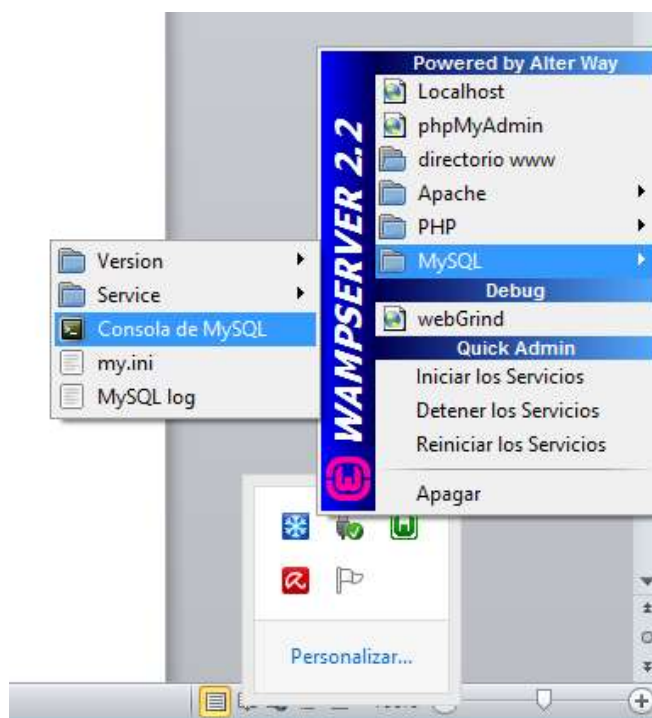


ACCESO A DATOS CON CODEIGNITER

El Acceso a datos en CodeIgniter se encuentra basado en un patrón denominado **ActiveRecord** que permite poseer una capa extra a la sección de abstracción de clases habitual. Esta nueva capa proporciona un API para realizar las operaciones típicas con la base de datos, tales como consultas, inserciones, modificaciones y borrados, sin necesidad de escribir ni una línea de SQL. La capa Active Record se encargará de ello por nosotros, generando código SQL sintácticamente correcto y adaptado al gestor de bases de datos que estemos utilizando en nuestro proyecto.

CONFIGURANDO LA CONEXIÓN A LA BASE DE DATOS.

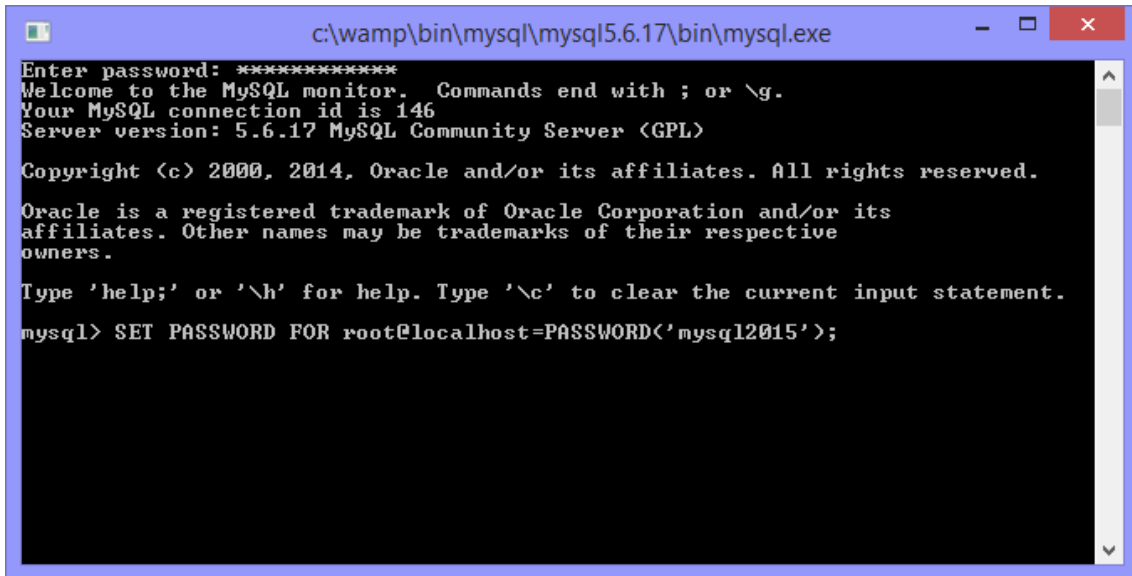
Los primeros pasos será configurar mysql que se encuentra en el wamp server, para ello damos click en el icono de wamp server y ejecutamos la consola de mysql como se ve en la imagen siguiente:



Si no se muestra esta configuración anteriormente, el usuario **root** no tiene configurada una contraseña, por tal motivo cuando se nos solicite el password solamente presionamos la tecla [ENTER]. Luego ingresamos la siguiente sentencia:

SET PASSWORD FOR root@localhost=PASSWORD('mysql2015');

Ejecutando dicho comando visualizaremos los resultados como aparecen a continuación en la siguiente imagen:



```
c:\wamp\bin\mysql\mysql5.6.17\bin\mysql.exe
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 146
Server version: 5.6.17 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SET PASSWORD FOR root@localhost=PASSWORD('mysql2015');
```

Luego de establecer la configuración del usuario administrador de nuestra base de datos, ya podemos establecer una conexión a ella con estas credenciales *usuario: root, password: mysql2015*.

Ahora creamos la siguiente base de datos:

```
CREATE DATABASE db_ids;
CREATE TABLE `persona` (
  `IdPersona` int(11) NOT NULL AUTO_INCREMENT,
  `Nombre` varchar(255) NOT NULL,
  `Apellido` varchar(255) NOT NULL,
  `Edad` varchar(255) NOT NULL,
  `Genero` char(1) NOT NULL,
  PRIMARY KEY (`IdPersona`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

CONFIGURACION DEL MODELO

En lugar de escribir operaciones de base de datos directamente en el controlador, las consultas deberían ubicarse en un modelo, para que se puedan reusar más tarde con facilidad. Los modelos son el lugar donde se devuelve, inserta y actualiza la información de la base de datos u otros almacenamientos.

Antes de iniciar la creación de un modelo dentro de nuestro proyecto es necesario que configuremos el archivo de configuración **database.php**, el cual nos permitirá el acceso a nuestra base de datos, en dicho archivo agregaremos el usuario, el password y la base de datos; la ubicación del archivo se encuentra en: **C:\wamp\www\ GUIA07\application\config** .

La configuración del archivo debería visualizarse de la siguiente manera:



```
47
48 $active_group = 'default';
49 $active_record = TRUE;
50
51 $db['default']['hostname'] = 'localhost';
52 $db['default']['username'] = 'root';
53 $db['default']['password'] = 'mysql2015';
54 $db['default']['database'] = 'db_ids';
55 $db['default']['dbdriver'] = 'mysql';
56 $db['default']['dbprefix'] = '';
57 $db['default']['pconnect'] = TRUE;
58 $db['default']['db_debug'] = TRUE;
59 $db['default']['cache_on'] = FALSE;
60 $db['default']['cachedir'] = '';
61 $db['default']['char_set'] = 'utf8';
62 $db['default']['dbcollat'] = 'utf8_general_ci';
63 $db['default']['swap_pre'] = '';
64 $db['default']['autoinit'] = TRUE;
65 $db['default']['stricton'] = FALSE;
66
67
68 /* End of file database.php */
69 /* Location: ./application/config/database.php */
```

Además de establecer la configuración de nuestro archivo **database.php**, configuraremos otro archivo llamado **config.php**, en dicho archivo configuraremos la sección **"base_url"**, esta sección nos ayudara para la carga del **helper** para las funciones de url, más adelante veremos que son los **helper**; la configuración del archivo **config.php**, debería visualizarse de la siguiente manera:

```
1 <?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2
3 /*
4  |-----
5  | Base Site URL
6  |-----
7  |
8  | URL to your CodeIgniter root. Typically this will be your base URL,
9  | WITH a trailing slash:
10 |
11 | http://example.com/
12 |
13 | If this is not set then CodeIgniter will guess the protocol, domain and
14 | path to your installation.
15 |
16 */
17 $config['base_url'] = 'http://localhost/Desarrollo/GUIA07';
18
```

Continuando con nuestro ejemplo crearemos nuestro modelo para poder insertar registro en la base de datos **db_ids**, agregaremos un nuevo modelo (Un nuevo archivo en la carpeta *models*) llamado **personas.php**, dentro de dicho archivo crearemos una clase que heredara de la **CI_Models**. El contenido del archivo será el siguiente:



```
1 <?php
2 class Personas extends CI_Model{
3     //Constructor de la clase
4     public function __construct()
5     {
6         $this->load->database();
7     }
8
9     //Funcion para obtener las personas de la base de datos
10    public function getPersonas()
11    {
12        $query = $this->db->get('persona');
13        return $query->result_array();
14    }
15
16    //Funcion para insertar una persona a la base de datos
17    public function createPersona()
18    {
19        //Obtenemos los datos
20        $data = array('Nombre'=> $this->input->post('nombre'),
21                     'Apellido'=> $this->input->post('apellido'),
22                     'Edad'=> $this->input->post('edad'),
23                     'Genero'=> $this->input->post('genero')
24                     );
25
26        //Insertamos la informacion
27        return $this->db->insert('persona',$data);
28    }
29 }
30
```

La función **database()**; es una función que es heredada de la clase **CI_Model**, esta función nos permite inicializar las funciones necesarias para el manejo y acceso a datos de nuestra base; ejecutada dicha función en el constructor de nuestra clase podemos utilizar funciones tales como:

- ***This->db->get('nombre tabla')***: Función que permite obtener los datos que se encuentre almacenados en la tabla que posee el nombre del parámetro, esto equivale a realizar una función que ejecute una consulta como ***SELECT * FROM [nombre Tabla]***.
- ***This->db->insert('nombre tabla', 'data array')***: Función que permite la inserción de información en nombre de la tabla que se pasa por parámetro, la información que será insertada es pasada como un ***data array*** , en el cual cada uno de los elementos del array representan los campos de las tablas con su respectivo valor.

FUNCIONES HELPER

Los helpers, como su nombre sugiere, lo ayudan con las tareas. Cada archivo de helper es simplemente una colección de funciones en una categoría particular. Hay Helpers de URL que ayudan en la creación de enlaces, Helpers de Formulario que lo ayudan a crear elementos de



formulario, Helpers de Texto que ejecutan varias rutinas de formateo de texto, Helpers de Cookie que escriben y leen cookies, Helpers de Archivo que ayudan a trabajar con archivos, etc.

A diferencia de la mayoría de otros sistemas de CodeIgniter, los Helpers no se escriben en formato Orientado a Objetos. Son simples funciones procedimentales. Cada función helper ejecuta una tarea específica, sin dependencia con otras funciones.

CodeIgniter no carga por defecto los Archivos Helper, por lo tanto el primer paso para usar un Helper es cargarlo. Una vez cargado, se hace disponible globalmente en su controlador y vista.

Creando el controlador.

Continuando con nuestro proyecto, crearemos un nuevo archivo en la carpeta **controller**, este archivo llevara como nombre **persona.php**, en este archivo agregaremos las funciones que atenderán a las peticiones del usuario para mostrar las vista y el insertado de datos. Un elemento importante que introduciremos es la inicialización de los **helpers** y las librerías para la validación del formulario. El código para el controlador será el siguiente:

```
1 <?php
2 class Persona extends CI_Controller{
3     //Constructor de la clase
4     public function __construct()
5     {
6         parent::__construct();
7         $this->load->helper('url');
8         $this->load->model('personas');
9         $this->load->library('form_validation');
10        $this->load->helper('form');
11    }
12
13    //Funciones para mostrar las vistas
14    public function index($view='index'){
15        //Obtenemos las personas
16        $data['personas'] = $this->personas->getPersonas();
17        $data['titulo'] = 'Personas del Sistema';
18
19        //validamos si existe la pagina
20        if (!file_exists('application/views/persona/'. $view. '.php')) {
21            show_404();
22        }
23
24        //Cargamos las vistas
25        $this->load->view('templates/header', $data);
26        $this->load->view('persona/'. $view, $data);
27        $this->load->view('templates/footer', $data);
28    }
29 }
```

La función para insertar la información de la persona seria la siguiente:



```
30 //Funcion para crear una persona
31 public function create() {
32
33     //Establecesmo las reglas de validacion
34     $this->form_validation->set_rules('nombre','Nombre','required');
35     $this->form_validation->set_rules('apellido','Apellido','required');
36     $this->form_validation->set_rules('edad','Edad','required');
37     $this->form_validation->set_rules('genero','Genero','required');
38
39     //Validamos que existan dichos campos
40     if($this->form_validation->run() === FALSE) {
41         $data['titulo'] = 'Personas del Sistema';
42         $this->load->view('templates/header', $data);
43         $this->load->view('persona/create', $data);
44         $this->load->view('templates/footer', $data);
45         echo "<script>alert('Datos Incorrectos');</script>";
46     } else {
47         $this->personas->createPersona();
48         redirect(base_url(). 'Index.php/persona/index');
49     }
50 }
51
52 }
53 }
```

La función `$this->form_validation->set_rules`, nos permite configurar las validaciones con respecto a los campos del formulario que nosotros especifiquemos para una validación. Los parámetros que son solicitados son los siguientes `$this->form_validation->set_rules(['Nombre Campo', 'Nombre Campo para visualizar en mensajes de error', 'Regla de validación'])`:

- **Nombre Campo:** Nombre del campo que es representado por un control en el formulario.
- **Nombre Campo para visualizar en mensajes de error:** Si el campo no cumple con las reglas de validación, se visualizara un mensaje mencionando el problema, dicho mensaje estará asociado al nombre del campo que coloquemos en este parámetro.
- **Regla de Validación:** En este parámetro colocaremos las reglas de validación para el campo que especifiquemos, podemos establecer varias validaciones, para concatenarlo utilizaremos el simbolo `"|"`, un ejemplo seria el siguiente: `required|min_length[5]|max_length[12]|is_unique[users.username]`.

Luego para ejecutar las configuraciones llamamos a la función `run`, de la siguiente manera: `$this->form_validation->run()`.

CREANDO LAS VISTAS.

Luego de crear nuestro controlador crearemos nuestras vistas, en la carpeta **view** agregaremos dos carpetas una llamada **persona y otra templates**. La distribución de los archivos seria los siguientes:

- **Persona:** Contendrá los archivos `create.php` , `index.php`.
- **Templates:** Contendrá los archivos `footer.php`, `header.php`.

Iniciamos con la creación de los archivos de vista para la carpeta **templates**, la codificación seria la siguiente:



Header.php

```
1 <html>
2 <head>
3     <title><?php echo $titulo ?></title>
4 </head>
5 <body>
6     <h1>Sistema de Implantacion de Sistemas</h1><br>
7     <h3><?php echo $titulo ?></h3>
8
```

footer.php

```
1 <strong>Universidad de Sonsonate &copy; 2013</strong>
2 </body>
3 </html>
```

Para la creación de los archivos de la carpeta personas iniciamos con el archivo *index.php*:

```
1 <?php if ($personas != null) {?>
2     <table border="1">
3         <tr>
4             <th>Id</th>
5             <th>Nombre</th>
6             <th>Apellido</th>
7             <th>Edad</th>
8             <th>Genero</th>
9         </tr>
10        <?php foreach ($personas as $persona) {?>
11            <tr>
12                <td><?php echo $persona['IdPersona'] ?></td>
13                <td><?php echo $persona['Nombre'] ?></td>
14                <td><?php echo $persona['Apellido'] ?></td>
15                <td><?php echo $persona['Edad'] ?></td>
16                <td><?php echo $persona['Genero'] ?></td>
17            </tr>
18        <?php } ?>
19    </table>
20    <button type="button" >Nuevo</button> | <button>Modificar</button> | <button>Eliminar</button><br />
21 <?php }else{ ?>
22     <p style="background: red;"><b>No existen Personas en el sistema!!</b></p>
23     <a href="index/create">Nueva Persona</a><br/>
24 <?php } ?>
```

Este archivo nos permitirá visualizar una tabla con el contenido de las personas almacenadas en la base de datos, dado el caso no se encuentre ninguna persona registrada se mostrará un mensaje ***"No existen Personas en el sistema"***.

La codificación del archivo *create.php* será el siguiente:



```
1 <h4>Agregar Nueva Persona</h4>
2 <?php echo validation_errors(); ?>
3 <?php echo form_open('persona/create') ?>
4 <fieldset>
5     <legend>Nueva Persona</legend>
6     <label>Nombre:</label><input type="text" name="nombre"/><br/>
7     <label>Apellido:</label><input type="text" name="apellido" /><br/>
8     <label>Edad:</label><input type="text" name="edad" /><br/>
9     <label>Genero:</label><br/>
10     <input type="radio" name="genero" value="m" />Masculino<br/>
11     <input type="radio" name="genero" value="f" />Femenino<br/>
12 </fieldset>
13 <input type="submit" name="submit" value="Crear Persona"/>
14
15 </form>
```

Ahora solamente nos falta acceder a las vista a través de la url, si deseamos ver el listado de las personas solamente tendríamos que colocar la siguiente dirección ***http://localhost/GUIA07/Index.php/persona/index***, para el caso de ingresar una nueva persona solamente cambiamos el parámetro index, por créate.

EJERCICIOS.

Tomar en cuenta la base de datos utilizada en la primer evaluación para aplicar el acceso a datos por medio de CodeIgniter para las 2 tablas.