CodeIgniter
Parte II: dinámico

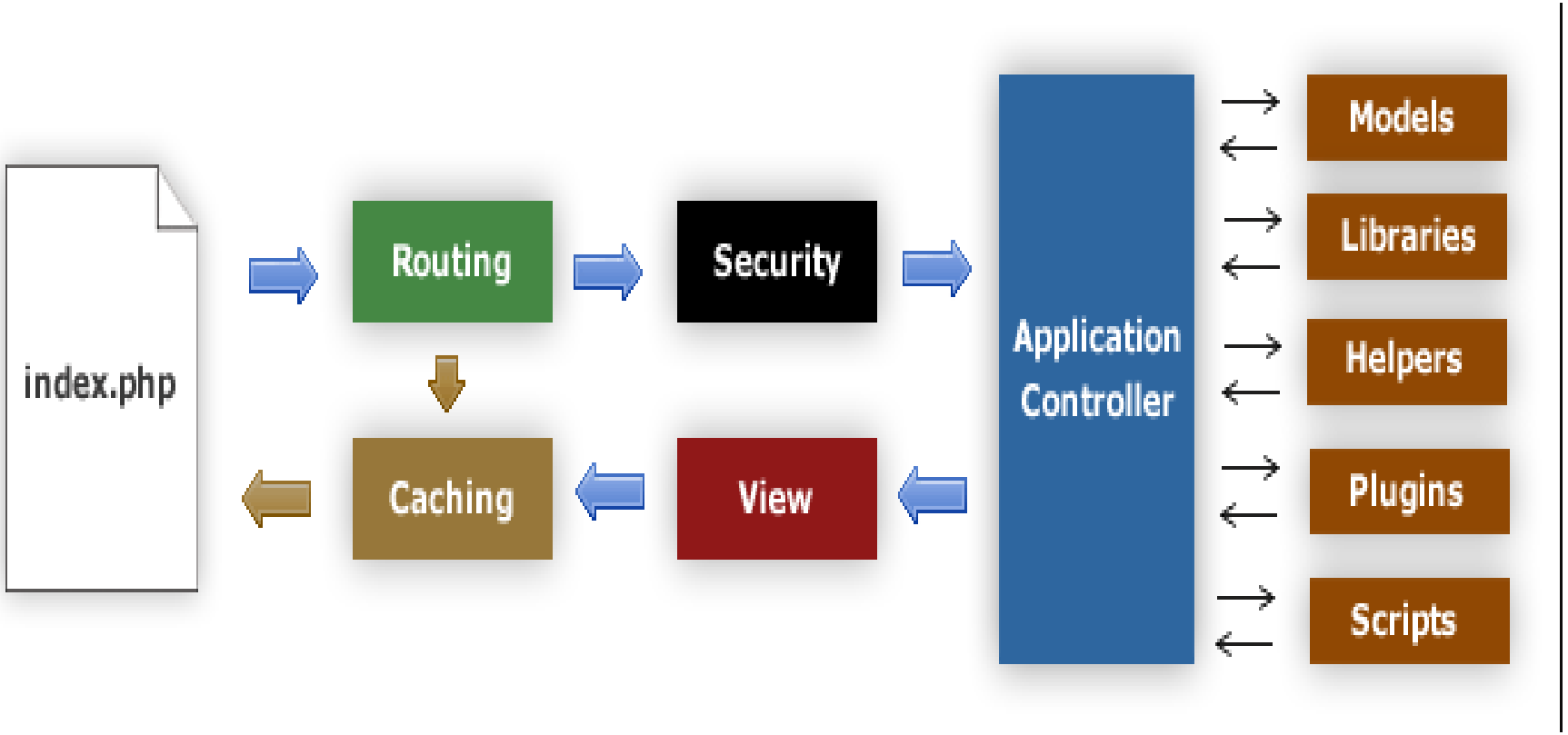
CodeIgniter

- Framework: una librería que proporciona un entorno de trabajo para un lenguaje de programación.
- Página web: https://www.codeigniter.com
- Utiliza arquitectura MVC: Modelo-Vista-Controlador.
- Permite crear páginas dinámicas utilizando la información extraída de la Base de Datos.

Modelo

- Representa las estructuras de datos.
- Las consultas a la base de datos se deben colocar en un modelo, de forma que puedan reutilizarse después.
- CodeIgniter incluye una capa de abstracción de la base de datos, Query Builder (https://www.codeigniter.com/userguide3/database/query_builder.html).
- Esto permite crear aplicaciones independientes de la tecnología de BD.

Diagrama de flujo en una aplicación



Configurar la Base de Datos

- Configura tu base de datos en el fichero application/config/database.php.

```
$db['default'] = array(
    'dsn'    => '',
    'hostname' => 'localhost',
    'username' => 'alumno',
    'password' => 'alumno',
    'database' => 'test',
    'dbdriver' => 'mysqli',
    'dbprefix' => '', …
```

- https://www.codeigniter.com/userguide3/installation/index.html

Configurar la Base de Datos II

- Aunque eso es suficiente la configuración puede ser muy compleja. Vale la pena mirar la documentación.

https://www.codeigniter.com/userguide3/database/configuration.html

- Cosas que se pueden determinar: casi todo. Por ejemplo, a que otras bases de datos/servidores conectarse si falla la actual.

Páginas dinámicas

- Las consultas **se deben colocar en el modelo**, no directamente en el controlador, para facilitar su reutilización.
- Los modelos son el lugar dónde se accede y modifica la información de la base de datos u otra fuente de datos. Representan tus datos.

Páginas dinámicas

- Se crea un nuevo modelo extendiendo CI_Model y cargando la librería de la base de datos.
- En el controlador:

```
$this->load->model('nombre_del_modelo');
```

- Esto hace a la BdD accesible a través del objeto $this->db

$this->nombre_del_modelo

Páginas Dinámicas: ejemplo

```
class News_model extends CI_Model {
   public function __construct()  {
        $this->load->database();        }

  public function get_news($slug = FALSE) {
        if ($slug === FALSE)    {
              $query = $this->db->get('news');
              return $query->result_array();
        }
        $query = $this->db->get_where('news', array('slug'
=> $slug));
        return $query->row_array();
  }   …  }
```

Páginas dinámicas: ejemplo model

```
class Blog_model extends CI_Model {
    public $title;      public $content;
    public function get_last_ten_entries()  {
        $query = $this->db->get('entries', 10);
        return $query->result();      }
    public function insert_entry()   {
        $this->title    = $this->input->post('title');
        $this->content  = $this->input->post('content');
        $this->db->insert('entries', $this);      }
    public function update_entry()  {
        $this->title    = $this->input->post('title');
        $this->content  = $this->input->post('content');
        $this->db->update('entries', $this, array('id' => $this-
>input->post('content')));
    } }
```

PD: ejemplo controller

```
class Blog_controller extends CI_Controller {
        public function index()  {
            ...
        }
        public function blog()  {
            $this->load->model('blog','',TRUE);
            $data['query'] = $this->blog-
>get_last_ten_entries();
            $this->load->view('blog', $data);
        }
}
```

PD: ejemplo view

```
<html>
<body> <h1>        Nueva entrada de blog     </h1>
   <?php  echo form_open('blog_controller/blog');
      echo form_label('Titulo', 'title');
      echo form_input('title'); echo '<br>';
      echo form_label('Contenido', 'content');
      echo form_input('content'); echo '<br>';
      echo form_submit('botonSubmit', 'Enviar');
      echo form_close();
      if(isset($query)) {
         echo "El título del primer blog es: ".$query[0]->title;
         echo "Y su contenido es: ".$query[0]->content;
      }   ?>
</body> </html>
```

PD: tarea ejemplo

- Adapta la vista del ejemplo para que muestre todos los datos devueltos por la consulta en una tabla.
- Crea un botón en la vista principal que te abra una vista que muestre la tabla (quita la tabla del formulario).
- Cambia el formulario para que al apretar submit se acabe llamando al método insert_entry() del modelo.

PD: tarea ejemplo

- Crea un método que te devuelva toda la tabla.
- Crea un método con un parámetro de entrada número que indique el número de filas a extraer de la tabla (y el controlador y la vista con formulario correspondiente).
- Vista y controlador para insertar una nueva fila.

Páginas dinámicas

- Se crea un nuevo modelo extendiendo CI_Model y cargando la librería de la base de datos. Esto hace a la BdD accesible a través del objeto $this->db

$this->load->database(); ó se conecta automáticamente (por ejemplo con la opción TRUE).

Conexión a la BdD

- Se puede conectar a varias bases de datos a la vez.

```
$DB1 = $this->load->database('group_one', TRUE);

$DB2 = $this->load->database('group_two', TRUE);
```

- Se pueden cambiar los parámetros de conexión especificados en el fichero de configuración.
- Se puede realizar cualquier acción.

Query builder: ejemplos de acceso

- Ejemplo “exótico”:

```
$data = array(
        'title' => $title,
        'name' => $name,
        'date' => $date );
$this->db->insert('mytable', $data);
// Produce: INSERT INTO mytable (title, name, date)
VALUES ('{$title}', '{$name}', '{$date}')
```

Query builder

- Muy complejo, leeros la lista de funciones que os he pasado.
- Ya hemos visto, sin embargo...

```
$query = $this->db->query('SELECT name, title
FROM my_table');

foreach ($query->result() as $row) {
        echo $row->title;
        echo $row->name;
}
```

Query builder: detalles

- En general los parámetros pueden pasarse como un array asociativo.
- Sentencias que ejecutan y otras que sólo preparan la consulta (_compiled_).
- Concatenado de métodos:

```
$this->db->select('title')->where('id',$id)->get_('mitabla');
```

Query builder: ejemplo clases

```
function mostrar_persona() {
   $this->load->database();
   $query = $this->db->query('SELECT nombre, dni
FROM persona');
  foreach ($query->result() as $row) {
       echo $row->nombre; echo " ";
       echo $row->dni;
       echo "<br>";
  }
  echo 'Total Results: ' . $query->num_rows();
} }
```

Query builder: ejemplo array

```
function mostrar_persona() {
   $this->load->database();
   $query = $this->db->query('SELECT nombre, dni
FROM persona');
  foreach ($query->result_array() as $row){
      echo $row['nombre'];   echo " ";
      echo $row['dni'];
      echo "<br>";
  }
  echo 'Total Results: ' . $query->num_rows();
} }
```

Query builder query: detalles

Importante:

- query() devuelve un objeto, TRUE o FALSE, no el resultado de la consulta, eso se obtiene a partir del objeto con result() o result_array().
- Esta pensada para usarse con INSERT, DELETE o UPDATE.
- Hay que controlar lo que le ponemos.

Ejemplo completo

- blog_model.php
- Blog_controller.php
- Blog.php (vista)
- Basados en la tabla **entries** (id, title, content) de la base de datos **test**.

Query builder query: error

Para controlar el error:

```
if ( ! $this->db->simple_query('
SELECT `example_field`
FROM `example_table`'))
{
        $error = $this->db->error();
        echo $error;
}
```

Query builder: resultados

Todo el resultado: result_object() (result()) y result_array().

Línea a línea: row_object() (row()) y row_array(). Además...

- $row = $query->row(numero_de_fila);
- $row = $query->first_row()
- $row = $query->last_row()
- $row = $query->next_row()
- $row = $query->previous_row()

Query builder: Tarea

Utilizar el método **get()** de Query builder para hacer lo mismo.

```
$query = $this->db->get('table_name');
```

¿Esto quiere decir que no hace falta SQL?

Query builder: select más complejo

```
function consulta_compleja() {
    $this->load->database();
    $query = $this->db->query('SELECT nombre,
dni FROM persona WHERE nombre="Gustavo
Casañ"');
    foreach ($query->result() as $row) {
        echo $row->nombre; echo " ";
        echo $row->dni;
        echo "<br>";
    } }
```

Tarea: probar otras condiciones en el WHERE

Query builder: Tarea

Utiliza métodos de Query builder para realizar la consulta anterior. Para empezar, ¿cuáles serían?

Ejercicio

- Crea un modelo para **mostrar** los datos de las personas en la tabla **persona** de la base de datos **test**. Crea el correspondiente controlador y vista y muéstralos en una tabla.
  - Nombre
  - DNI
  - Dirección
  - Fecha de Nacimiento
  - Teléfono
  - Email
- Tarea extra: utiliza css para darle buen aspecto.

Ejemplo: insert

```
$data = array(
        'title' => 'My title',
        'name' => 'My Name',
        'date' => 'My date'
);

$this->db->insert('mytable', $data);
```

Ejercicio:

- Crea un formulario para **introducir** los datos de una persona, validarlos e introducirlos en la tabla **persona** de la base de datos **test**:
  - Nombre
  - DNI
  - Dirección
  - Fecha de Nacimiento
  - Teléfono
  - Email

Ejemplo: actualizar

```
$this->db->set('name', $name);
$this->db->set('title', $title);
$this->db->set('status', $status);
$this->db->where('id', 2);
$this->db->update('mytable');
```

Ejercicio

- Crea un formulario para **modificar** los datos de una persona, validarlos e introducirlos en la tabla **persona** de la base de datos **test**:
  - Nombre
  - DNI
  - Dirección
  - Fecha de Nacimiento
  - Teléfono
  - Email

Ejemplo: varias tablas

```
function get_categories($id) {
        $this->db->select('*');
        $this->db->from('entries_categories');
        $this->db-
>where('entries_categories.entry_id ='.$id);
        $this->db->join('entries', 'entries.id =
entries_categories.entry_id');
        $this->db->join('categories', 'categories.id =
entries_categories.category_id');
        $this->db->order_by("category", "asc");
   return $this->db->get()->result();        }
```

Ejemplo: varias tablas

```
$this->db->select('u.*, c.company, r.description');
    $this->db->from('users u, company c, roles r');
    $this->db->where('c.id = u.id_company');
    $this->db->where('r.permissions = u.permissions');
    $query = $this->db->get();
```

Ejercicio

- Crea un método que dada una persona, consulte las tablas **persona, ventas** y **productos** de la base de datos **test** (aseguraros que están relacionadas). Crea controlador y vista para, en una tabla, mostrar los siguientes campos:
  - Nombre de la persona
  - Fecha de la compra
  - Precio final cantidad*precio-descuento+IVA
  - Nombre del producto

Ejercicio: tenis

- En un ejercicio anterior creaste una página sobre el tenis. Ahora crea la base de datos adecuada (tabla usuarios con sus campos y tabla cargos con los campos id, nombre, cargo, fecha_inicio y fecha_final, que puede ser nulo) y conéctate a ella con CodeIgniter.
- Se deberá poder crear un nuevo usuario, comprobar si el usuario existe y dejarle acceder a otras páginas y mostrar la tabla de cargos a partir de la información en la BdD.

CI: subconsultas

- CodeIgniter NO contempla las subconsultas. Como hay versiones de SQL que no las permiten, ellos tampoco. Es decir, NO hay un método $this->db->subquery()
- Sin embargo, se pueden utilizar de una forma más o menos indirecta.

```
$query=$this->db->query('SELECT * FROM productos WHERE id NOT IN (SELECT productoID FROM ventas)');
```

CI: subconsultas

```
public function subconsulta() {
    $sub_query_from = '(SELECT id, product FROM product ) as product';
    $this->db->select();
    $this->db->from($sub_query_from);
    $query = $this->db->get();
    return $query->result();
}
```

CI: subconsultas

```
public function subconsulta() {
        $subquery = 'SELECT productoID FROM
ventas';
        $this->db->select('*');
        $this->db->where_not_in('id', $subquery);
        $this->db->from('productos');
        $query = $this->db->get();
        return $query->result();
    }
```

Tarea: comprueba que esto funciona mostrando los resultados en una vista.

CI: subconsultas

```
public function subconsulta() {
        $subquery = 'SELECT productoID FROM
ventas';
        $this->db->select('*');
        $this->db->where_not_in('id', $subquery);
        $this->db->from('productos');
        $query = $this->db->get();
        return $query->result();
    }
```

Tarea: comprueba que esto funciona mostrando los resultados en una vista.

Convertir estas consultas a QB I

- INSERT INTO coleccion(nombre, precio, IVA) VALUES ('Literatura', '12.45', '1.34');
- SELECT CONCAT(a.nombre, " " ,a.apellidos) AS Autor, " es el autor de ", l.titulo AS Titulo FROM autor a, libro l, autor_libro au WHERE a.id=au.id_autor AND l.id=au.id_libro ORDER BY 1, l.titulo DESC
- SELECT l.titulo, CONCAT (a.nombre, " ", a.apellidos) AS Autor, FORMAT((c.precio + 10 + ROUND(RAND() * 10)),2) AS "Precio Final" FROM autor a, libro l, autor_libro au, coleccion c WHERE a.id=au.id_autor AND au.id_libro=l.id AND l.id_coleccion=c.id

Convertir estas consultas a QB II

- SELECT p.descripcion, "No activa" FROM promocion p WHERE p.est_activo=0

  UNION

  SELECT p.descripcion, l.titulo FROM libro l, promocion p WHERE l.id_promocion=p.id AND p.est_activo=1

Realizar estas consultas en QB II

- Para la base de datos ENI, crea una función que nos devuelva los nombres de los autores y los temas que tienen sus libros.
- Para la base de datos ENI, crea una función que nos devuelva las promociones y los títulos de los libros que pertenecen a ellas. Y ya puestos, el precio y el IVA de cada libro (tendrás que cogerlo de la tabla coleccion).