

# Framework Semilla

¿Por qué? la propuesta

Cambios de paradigmas de programación, impactando en la productividad y eficiencia del desarrollo de las aplicaciones, permitiendo generar código, ajustándonos a las nuevas técnicas de diseño de sistemas, creando separadamente los archivos, con un instalador que nos permite hacer todo desde tan sólo 9 pasos para ver un sistema funcionando.

# Framework Semilla

## Ventajas de la propuesta

- 1.- Reducción en los tiempos de respuestas de 3 mese a 3 minutos
- 2.- Mejor metodología de implementación de patrones de diseño
- 3.- Abstracción de los distintos lenguajes
- 4.- Separación del código
- 5.- Renderización de las vistas

# Framework Semilla

## Notas Previas

Es importante tener en cuenta que el “alma” de los sistemas es su base de datos, siguiendo estas recomendaciones podremos hacer mejor uso del framework semilla

Tablas:

```
CREATE TABLE tbl_estatus_constancias (  
  id_estatus_constancia serial NOT NULL,  
  estatus_constancia character varying(80) NOT NULL,  
  CONSTRAINT pk_id_estatus_constancia PRIMARY KEY (id_estatus_constancia)  
)
```

# Framework Semilla

## Notas Previas

```
CREATE TABLE tbl_constancias (  
  id_constancia serial NOT NULL, fecha_solicitud date NOT NULL, fecha_entrega date,  
  observacion character varying(250), cedula_personal integer NOT NULL,  
  id_tipo_constancia integer NOT NULL, id_condicion integer NOT NULL,  
  id_estatus_constancia integer NOT NULL, cantidad integer NOT NULL DEFAULT 1,  
  CONSTRAINT pk_id_constancia PRIMARY KEY (id_constancia),  
  CONSTRAINT condicion FOREIGN KEY (id_condicion)  
    REFERENCES tbl_condicion (id_condicion) MATCH SIMPLE  
    ON UPDATE CASCADE ON DELETE RESTRICT,  
  CONSTRAINT fk_cedula_personal_constancia FOREIGN KEY (cedula_personal)  
    REFERENCES tbl_personal (cedula_personal) MATCH SIMPLE  
    ON UPDATE CASCADE ON DELETE RESTRICT,  
  CONSTRAINT tipo_constancia FOREIGN KEY (id_tipo_constancia)  
    REFERENCES tbl_tipo_constancia (id_tipo_constancia) MATCH SIMPLE  
    ON UPDATE CASCADE ON DELETE RESTRICT) ,  
  CONSTRAINT estatus_constancia FOREIGN KEY (id_status_constancia)  
    REFERENCES tbl_estatus_constancias (id_status_constancia) MATCH SIMPLE  
    ON UPDATE CASCADE ON DELETE RESTRICT)
```

# Framework Semilla

## Notas Previas

Las tablas llamadas en plural, los campos identificadores y Primary Key llamarse id\_nombre\_tabla en singular, los campos descripción evitarlos y convertirlos en nombre de la tabla.

Ejemplo:

[antes]

```
CREATE TABLE tbl_tipos_documentos (  
  id serial NOT NULL,  
  descripcion character varying(80) NOT NULL,  
  CONSTRAINT pk_id_tipo_documento PRIMARY KEY (id_tipo_documento) )
```

[ahora]

```
CREATE TABLE tbl_tipos_documentos (  
  id_tipo_documento serial NOT NULL,  
  tipo_documento character varying(80) NOT NULL,  
  CONSTRAINT pk_id_tipo_documento PRIMARY KEY (id_tipo_documento) )
```

Creado por Freddy Peñalver versión 2

# Framework Semilla

## Notas Previas

Todo esto se recomienda por que el Framewok semilla, mapea la base de datos y a partir de la estructura de las tablas creará los archivos en disco: formularios, archivos js, controladores y clases.

Esto nos permitirá construir mejores consultas (querys), ya que podremos implementar “using”

ejemplo:

```
select * from tbl_constancias
```

```
join tbl_estatus_constancias using(id_estatus_constancia)
```

# Framework Semilla

## Aclaratoria

Este documento esté orientado a desarrolladores que ya tienen experiencia con el lenguaje de programación PHP, significa que no vamos a cubrir temas realmente básicos como tipos de datos, variables, estructuras de control y similares. Debes saber acerca de estos tópicos para entender este documento.

# Framework Semilla

## Introducción

Se implementan las siguientes arquitecturas de software y patrones de diseño:

- MVC
- Singleton
- ORM
- POO
- Comportamiento – configuración
- Abstracción de base de datos
- Plantillas -- layouts -- Renderizar



# Framework Semilla

Librerías, Framework y plugin utilizados:

- ADObd
- JQuery
- FPDF
- Alphanumeric
- Marketo
- jquery.validate

# Framework Semilla

## Estructura de carpetas

- Controlador
- Documentación
- Inc
- instalar
- Js
- Log
- Modelo
- Prueba
- Vista
- Vista/layouts

# Framework Semilla

## Estructura de carpetas

- **Controlador:** contiene la lógica del negocio, según la solicitud del usuario realizará sobre el modelo una determinada acción en la base de datos
- **Documentación:** contendrá toda la documentación del sistema, tanto técnica como para el usuario final
- **Inc:** contendrá la configuración del sistema, con los archivos ***config.sistema, head y controlador\_frontal***
- **Instalar:** contiene el instalador del Framework Semilla

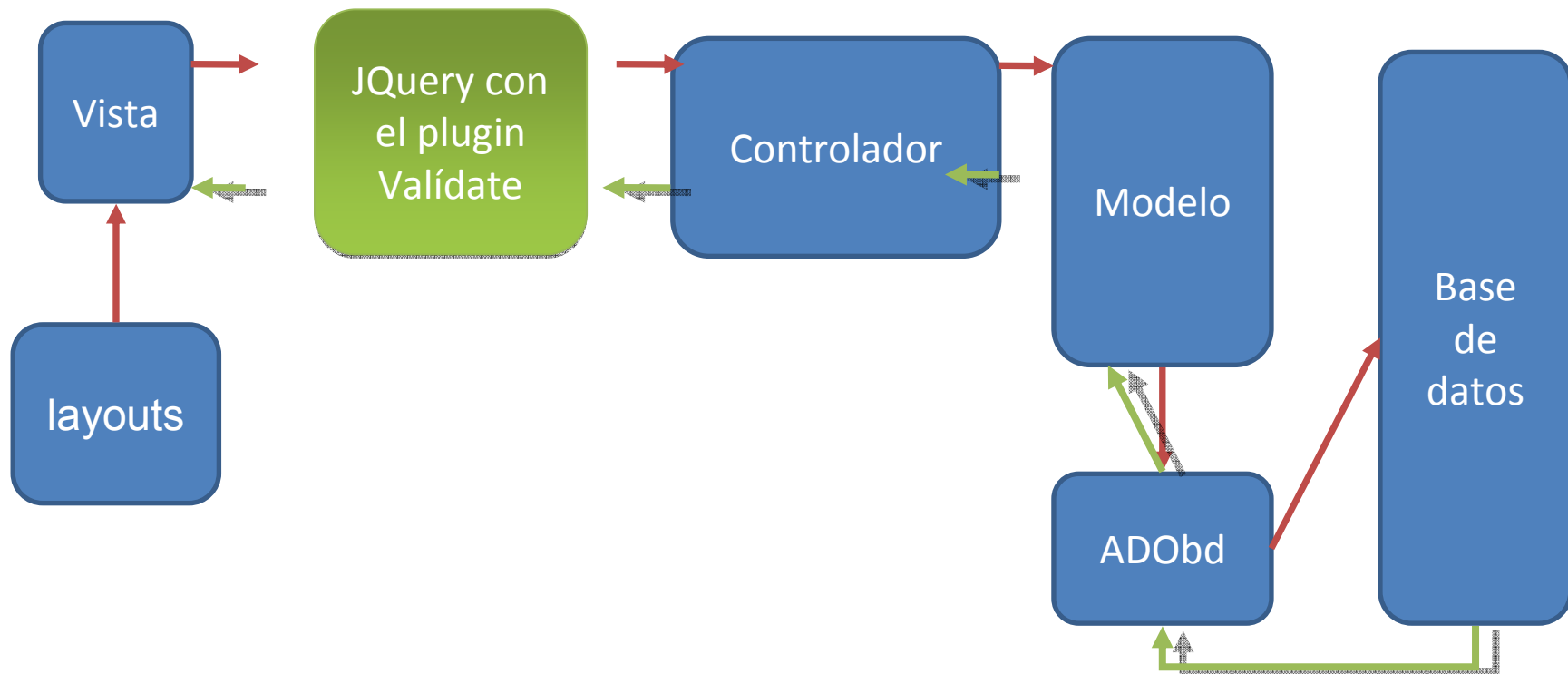
# Framework Semilla

## Estructura de carpetas

- **Js:** contendrá todos los archivos de Java Script, los framework y plugin a ser utilizados en el sistema
- **Log:** es el log del sistema generado por ADObd según se genere un problema con la base de datos
- **Modelo:** contendrá todas las clases del sistema y las librerías escritas en php a ser implementadas
- **Prueba:** es para realizar cualquier prueba de componentes como un laboratorio de trabajo.
- **Vista:** contendrá los formularios y vistas de cada tabla de la base de datos

# Framework Semilla

## Esquema de Funcionamiento



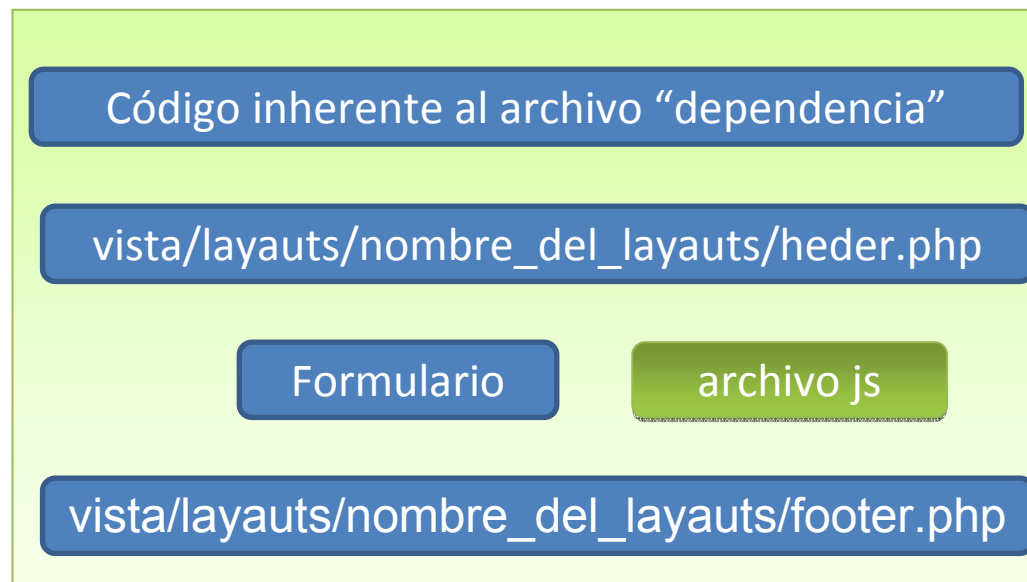
# Framework Semilla

## Acoplamiento

La estructura de la carpeta es vista/nombre de tabla/formulario.php

ejemplo: vista/tbl\_usuarios/formulario.php

El archivo formulario.php



# Framework Semilla

layouts

## Acoplamiento

La estructura de la carpeta es vista/layouts/nombre del layout  
ejemplo: vista/layouts/menpet

Contenido: css, imágenes, js (propios del layouts)

Estas carpetas son las propias del layouts, es importante tener los archivos header.php y footer.php, dentro de todos los layouts, ya que estos son los que harán la renderización de las vistas, de igual modo podemos incluir los archivos que deseemos para hacer mas atractivo nuestro layouts

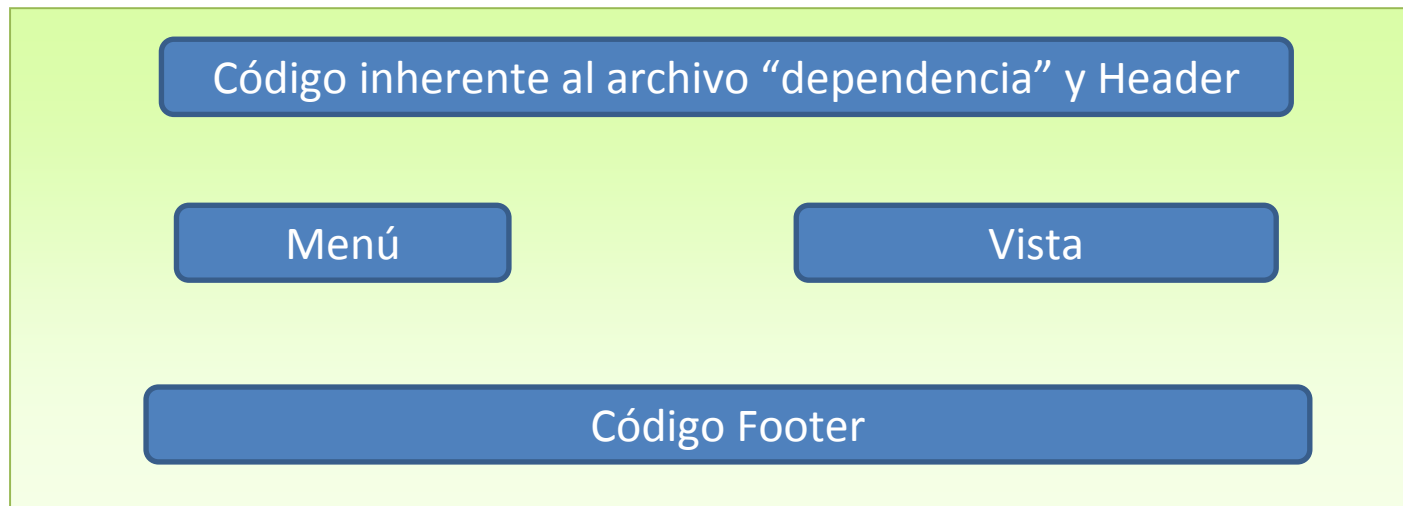
# Framework Semilla

layouts

## Acoplamiento

La estructura de la carpeta es vista/layouts/nombre del layout  
ejemplo: vista/layouts/menpet

Contenido: css, imágenes, js (propios del layouts)





JQuery con  
el plugin  
Valídate

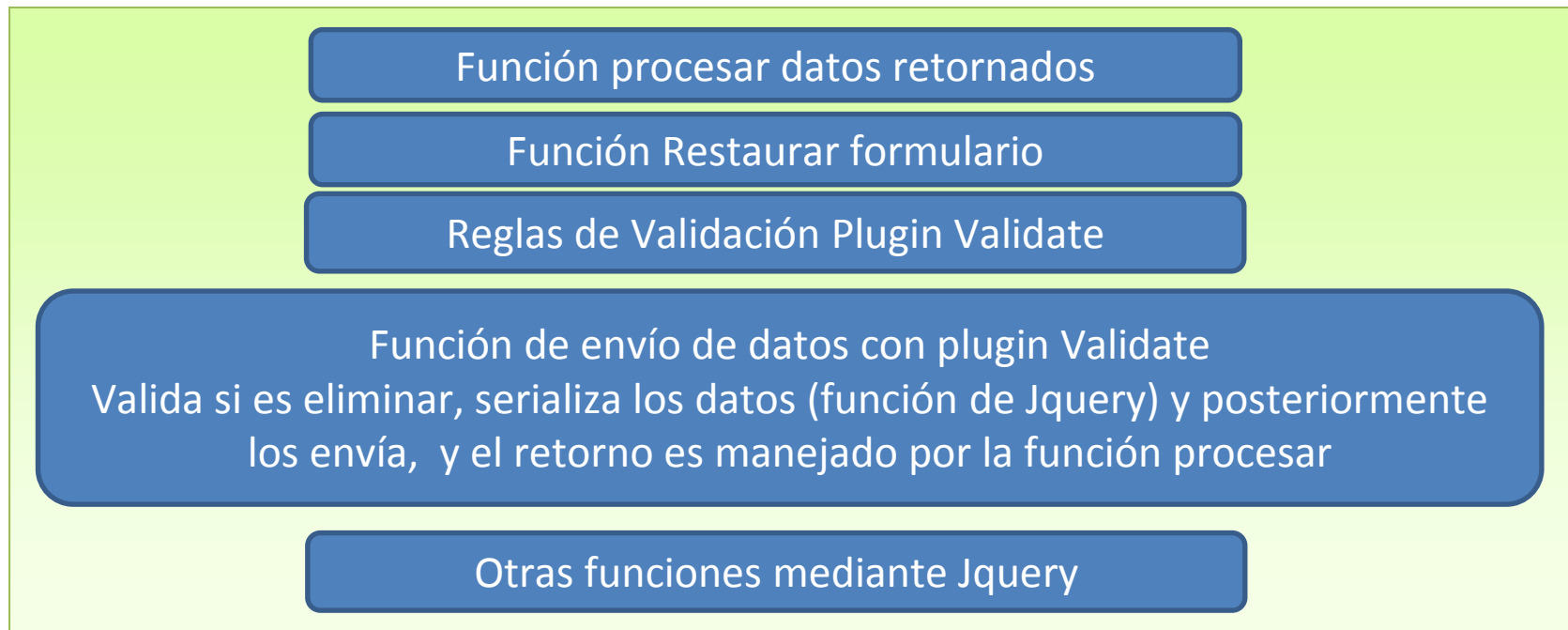
# Framework Semilla

## Acoplamiento

La estructura de la carpeta es js/validaciones/nombre de tabla.js

ejemplo: js/validaciones/tbl\_usuario.js

El archivo tbl\_usuario.js



Creado por Freddy Peñalver versión 2

JQuery con  
el plugin  
Valídate

# Framework Semilla

## Importancia

Lo mas importante de estos archivos es saber enviar los datos y recibirlos, con la atomicidad y rapidez de los formatos “JSON” o “JSONP”, según sea el caso, estos archivos son los que establecen la comunicación asincrónica, entre la Vista y el Controlador, pueden pre procesar información, antes del envío o posterior al recibirlos.

La curva de aprendizaje de esta librería es bastante rápida y la implementación de los plugin están bien documentadas, ya que es un framework de desarrollo, su documentación esta bien distribuida por Internet y en caracas existen varias compañías que dictan cursos

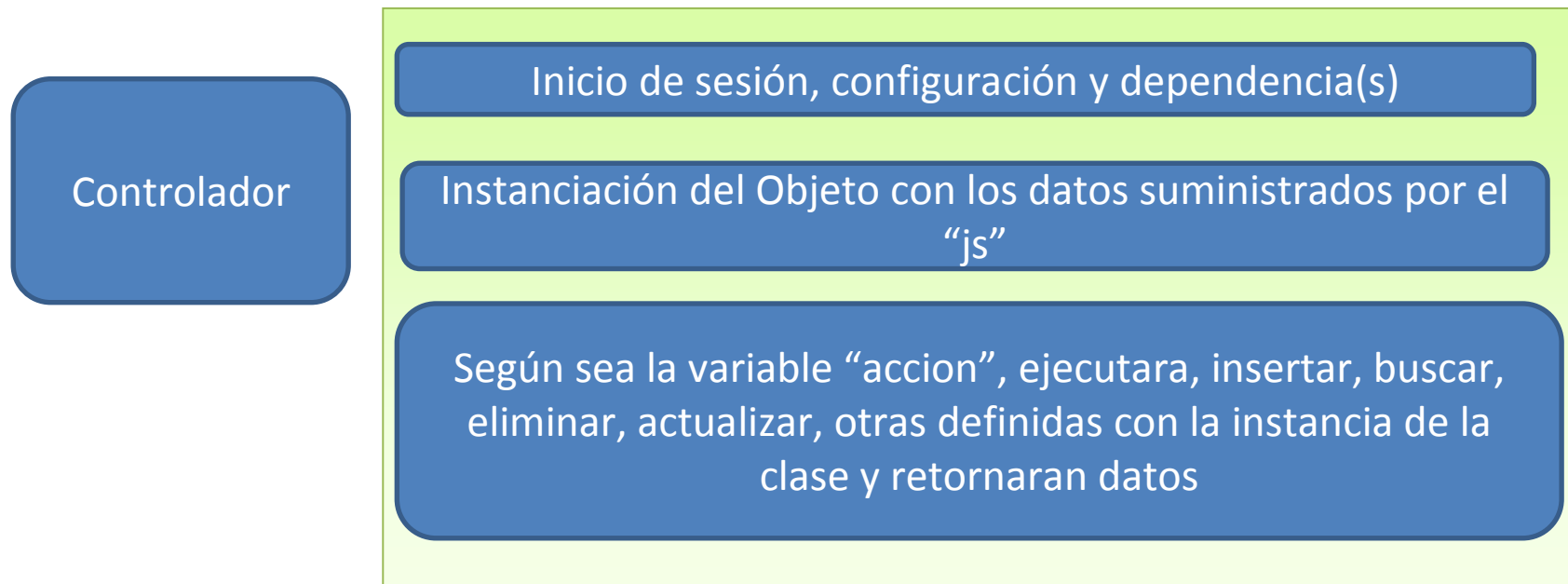
# Framework Semilla

## Acoplamiento

La estructura de la carpeta: controlador/nombre de tabla.php

ejemplo: controlador/tbl\_usuario.php

El archivo tbl\_usuario.php



# Framework Semilla

## Esquema de Funcionamiento

La creación de las clases es todo un conjunto de  
Métodos y Buenas Practicas de Programación:

1. Herencia de Clases
2. Clases Abstractas
3. Patrón de Diseño Singleton en Clases Main y BD
4. ORM (Objetos Relacional Mapeado)

# Framework Semilla

5. Definición de Objeto SQL
6. Implementación de Métodos Mágicos del lenguaje PHP
7. Funciones de insertar, actualizar con Librería ADOdb directamente
8. Mayor Abstracción de SQL dentro de la clase

# Framework Semilla

1. Herencia de Clases: existe una clase denominada `class_general.php` es donde están los Métodos y Atributos para todos los objetos de uso general
2. Clases Abstractas para la implementación del Patrón de diseño Singleton
3. Patrón de Diseño Singleton en Clases Main y BD: ya que todos los objetos al ser contruidos implementaran la misma instancia de inicio y base de datos, para no crear una conexión por cada objeto, permitiendo crear log de BD

# Framework Semilla

4. ORM (Objetos Relacional Mapeado): otros escritores le denominan objetos persistentes con la base de datos, es simplemente llevar los atributos de la tabla a variables dentro de un array
5. Definición de Objeto SQL: Para mayor abstracción del lenguaje SQL se creo otra clase para definir los Atributos públicos
- 6.- Implementación de Métodos Mágicos del lenguaje PHP, para mi es una de las formas mas rápidas y flexibles de integración de clases y la librería ADOdb, ya que permite definir que hacer cuando no exista un atributo o un método dentro de la clase

# Framework Semilla

7. Funciones de insertar, actualizar con Librería ADOdb directamente: esta librería permite hacer los SQL de una forma automática y permite escalar la aplicación a solo su utilización logrando la mayor abstracción de la Base de Datos

8. Mayor Abstracción de SQL dentro de la clase: la ventaja de crear SQL indispensables dentro de clase reduciéndolo al método listar()



Modelo

# Framework Semilla

## Acoplamiento

La estructura de la carpeta es  
modelo/class\_nombre\_de\_tabla.js

ejemplo: modelo/class\_tbl\_usuarios.php

El archivo class\_tbl\_usuario.php

Star session

Clase heredada de la general

Atributos Públicos y privados

Métodos Mágicos : \_\_construct, \_\_set, \_\_get, \_\_call

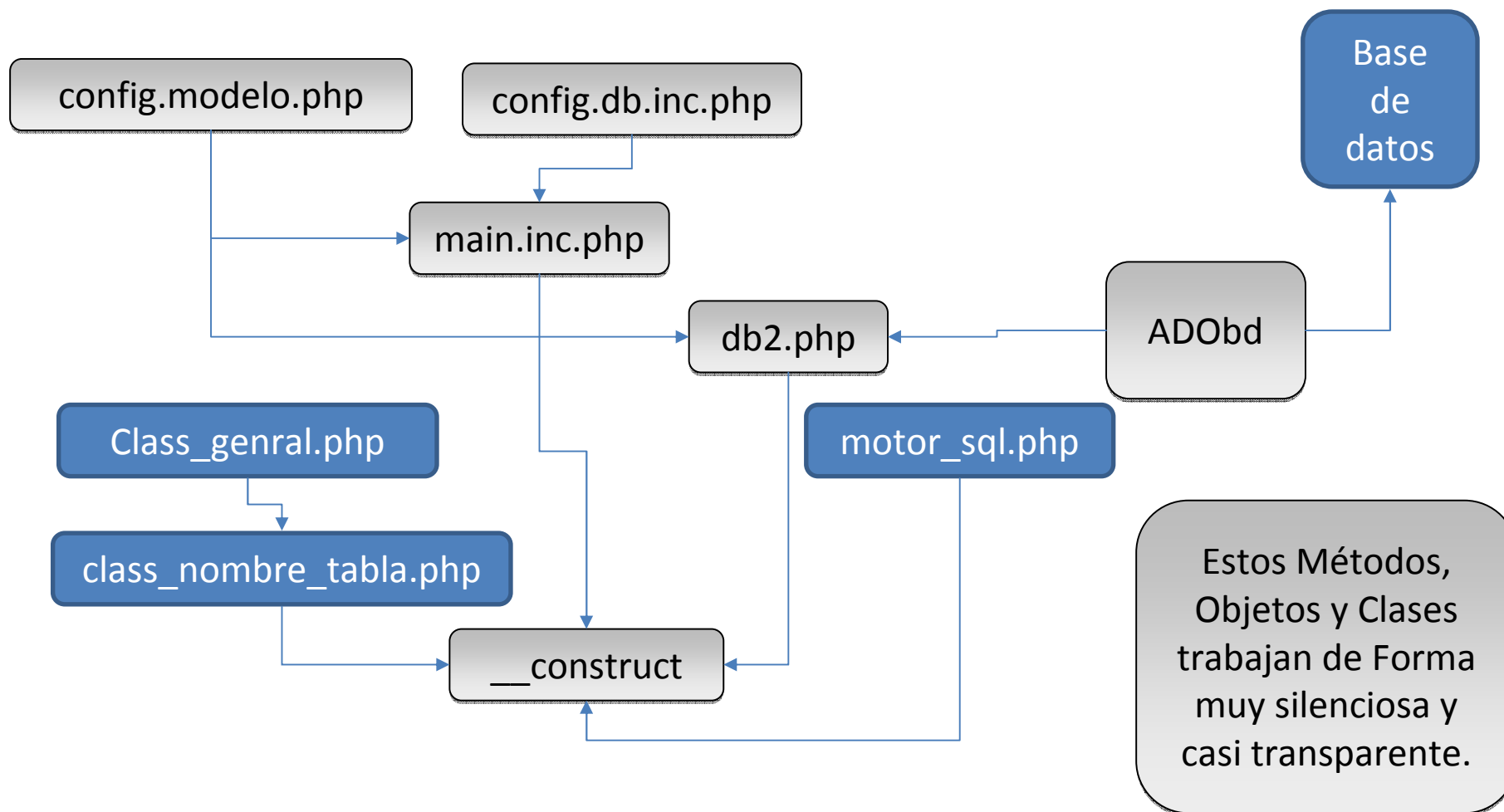
Métodos Públicos: insertar(), actualizar(), eliminar(), listar(), limit(), buscar(),  
ver\_opciones\_nombre\_tabla(), valor\_campo(), otras definidos según necesidades

En todos los métodos se implementa la librería ADObd y el manejo de excepciones  
(try / catch)

Modelo

# Framework Semilla

## Esquema de Funcionamiento



Creado por Freddy Peñalver versión 2

Modelo

# Framework Semilla

config.modelo.php  
archivo que contiene  
la configuración de las  
clases

config.db.inc.php  
archivo que tiene las  
variables de conexión a la  
base de datos

ADObd  
Librería de  
trabajo

Base  
de  
datos

main.inc.php  
Archivo que depende de  
config.db.inc.php que implementa  
singleton para todas las clases

db2.php  
archivo que implementa singleton para todas las  
clases cuando se instancias solo manejan una  
sola conexión a la base de datos

class\_nombre\_tabla.php la clases  
por cada tabla de la base de datos

\_\_construct método que se inicia de forma  
transparente cuando es instanciada la clase

class\_genral.php  
Archivo que contiene métodos y  
atributos generales

motor\_sql.php archivo definición de las  
particularidades del SQL según el Motor

# Framework Semilla

## ¿Cómo Instalar?

1.- copiar la carpeta freamework\_semilla, cambiar el nombre según el sistema:

ejemplo: **sigli**

2.- ejecutar el navegador con la dirección del servidor y la carpeta ejemplo:

<http://localhost/sigli/instalar/>

3.- seguir los 7 pasos de instalación, que se muestran una vez ejecutado el paso anterior

# Framework Semilla

## Posterior a la Instalación

4.- cambios posteriores en el caso postgres:

En la carpeta modelo, en el archivo `class_general.php` editar la variable:

`$this->schema="schema_base_de_datos",`  
según sea la mas utilizada para las clases, por otro lado si tenemos varios schemas, en cada clase se puede reescribir dicha variable en el constructor propio de cada clase

# Framework Semilla

¿Cómo hacer renderización?

Simple en el archivo `inc/config.sistema.php`, modificar la variable **`$default_layouts`** por el nuevo nombre del layouts, que es el nombre de la carpeta dentro de `/vista/layouts/`

Ejemplos:

```
$default_layouts = "default";
```

```
$default_layouts = "menpet";
```

```
$default_layouts = "menu_superior";
```

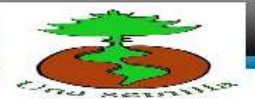
# Framework Semilla

Instalador Web Framework Semilla

localhost/sistema\_biblioteca\_2/instalar/config\_sist.php

Google


## Semilla Instalación



Previo Siguiente

### Pasos

- 1 : Bienvenida
- 2 : Licencia
- 3 : Configuración Aplicacion
- 4 : Base de datos
- 5 : Framework Semilla
- 6 : Menu
- 7 : Finalizar



### Configuración Aplicacion

Configuración Aplicacion

Configuración de la Aplicacion

Servidor	<input type="text" value="http://localhost"/>	<i>recuerde colocar http://nombre_ser_servidor</i>
Carpeta Raiz del sistema	<input type="text" value="sistema_biblioteca_2"/>	<i>recuerde no colocar espacios</i>
nombre del sistema	<input type="text" value="nombresistema"/>	<i>este nombre sera utilizado como titulo de la aplicacion</i>
Layout por defecto para toda la aplicacion	<input type="text" value="default"/>	<i>es recomendable dejar esta seccion con el valor de defecto si no cuenta con otro layout</i>

framework semilla! es software libre distribuido bajo la licencia GNU/GPL.

Creado por Freddy Peñalver versión 2

# Framework Semilla

## Pensamientos

- *Si quieres hacer realidad lo imposible, simplemente pateas la “im” y conviértelo en “posible”. (pensamiento SCOUT)*
- *Dame una palanca lo suficientemente fuerte y un punto de apoyo... y moveré la tierra. (Arquimedes)*
- *Las oportunidades se multiplican a la medida que son aprovechadas. (Sun Tzu)*
- *La locura está en pretender obtener un resultado diferente, haciendo siempre lo mismo. (Albert Einstein)*
- *Si quieres cambiar al mundo simplemente HAZLO!! F.:P.:.*

• *Gracias por se parte de esto!!*