Programación de 3 capas

La **programación por capas** es un modelo de desarrollo de *software* en el que el objetivo primordial es la separación (desacoplamiento) de las partes que componen un sistema de *software*: lógica de negocios, capa de presentación y capa de datos. De esta forma, por ejemplo, es sencillo y mantenible crear diferentes interfaces sobre un mismo sistema sin requerirse cambio alguno en la capa de datos o lógica.

CAPAS

Capa de presentación: la que ve el usuario (también se la denomina «capa de usuario»), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser «amigable» (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio. **La llamaremos capa fachada para efectos de este curso.**

Capa de negocio: Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él.

De ahora en adelante la llamaremos DAO. El DAO encapsula el acceso a la base de datos. Por lo que cuando la capa de lógica de negocio necesite interactuar con la base de datos, va a hacerlo a través de la API que le ofrece DAO. Generalmente esta API consiste en métodos CRUD (Create, Read, Update y Delete). Entonces por ejemplo cuando la capa de lógica de negocio necesite guardar un dato en la base de datos, va a llamar a un método create(). Lo que haga este método, es problema de DAO y depende de como DAO implemente el método create(), puede que lo implemente de manera que los datos se almacenen en una base de datos relacional, como puede que lo implemente de manera que los datos se almacenen en ficheros de texto. Lo importante es que la capa de lógica de negocio no tiene porque saberlo, lo único que sabe es que el método create() va a guardar los datos, así como el método delete() va a eliminarlos, el método update() actualizarlos, etc. Pero no tiene idea de como interactúa DAO con la base de datos.

Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. **Ahora llamaremos a la capa de datos DTO** y éstos son utilizados por DAO para transportar los datos desde la base de datos hacia la capa de lógica de negocio y viceversa.

Por ejemplo, cuando la capa de lógica de negocio llama al método create_user()

¿qué es lo que hace DAO? inserta un nuevo dato…

¿pero qué dato? el que la fachada le pase como parámetro…

¿y cómo se lo pasa este dato? bueno, a través de un DTO.

Ejemplos:

En la siguiente imagen se muestra un ejemplo de un DAO en PHP (clase), el dao contiene funciones o mejor conocidas como métodos, el ejemplo siguiente sólo tiene el elemento Read del CRUD en una tabla.

El método se llama: regresaUsuarios():

```
class daoUsuarios{
    public function regresaUsuarios($idNegocio){
    $sql='SELECT ID_USUARIO, NOMBRE, FECHA_ENTRADA, PASSWORD, ID_NEGOCIO, '.
            ' FECHA_BAJA, EMAIL_USUARIO, PWD_CORREO, FECHA_VIGENCIA, '.
            ' MAX_CLIENTES '.
            ' FROM LB_C_USUARIO  WHERE ID_NEGOCIO ='.$idNegocio. ' AND FECHA_BAJA IS NULL;';
    $campos=array("ID_USUARIO"=>"",
        "NOMBRE"=>"",
        "FECHA_ENTRADA"=>"",
        "PASSWORD"=>"",
        "ID_NEGOCIO"=>"",
        "FECHA_BAJA"=>"",
        "EMAIL_USUARIO"=>"",
        "PWD_CORREO"=>"",
        "FECHA_VIGENCIA"=>"",
        "MAX_CLIENTES"=>"");
    DBConexion::consulta($sql,$campos);
    $datos=DBConexion::$results;
    $i= 0;
    $Larray = array();
    foreach($datos as $dato){
            $miUsuario = new dtoUsuario();
            $miUsuario->setId_usuario     (isset(   $dato["ID_USUARIO"])    ?          $dato["ID_USUARIO"]       : -1);
            $miUsuario->setNombre         (isset(   $dato["NOMBRE"])        ?strtoupper($dato["NOMBRE"])         : "NONAME");
            $miUsuario->setFecha_entrada  (is_null( $dato["FECHA_ENTRADA"]) ?          NULL                          : $dato["FECHA_ENTRADA"]);
            $miUsuario->setPassword       (isset(   $dato["PASSWORD"])      ?          $dato["PASSWORD"]         : "PWD");
            $miUsuario->setId_negocio     (isset(   $dato["ID_NEGOCIO"])    ?          $dato["ID_NEGOCIO"]       : 0);
            $miUsuario->setFecha_Baja     (is_null( $dato["FECHA_BAJA"])    ?          NULL                          : $dato["FECHA_BAJA"]);
            $miUsuario->setEmail_usuario  (isset(   $dato["EMAIL_USUARIO"]) ?          $dato["EMAIL_USUARIO"]    : "@");
            $miUsuario->setPwd_correo     (isset(   $dato["PWD_CORREO"])    ?          $dato["PWD_CORREO"]       : "PWD-MAIL");
            $miUsuario->setFecha_vigencia(isset(   $dato["FECHA_VIGENCIA"])?          $dato["FECHA_VIGENCIA"]   : "0");
            $miUsuario->setMax_clientes   (isset(   $dato["MAX_CLIENTES"])  ?          $dato["MAX_CLIENTES"]     : 0);
            $Larray[$i] = $miUsuario;
            $i= $i + 1;
    }
    return $Larray;
}
```

Práctica:

En la práctica anterior se creo un clase llamada dtoDeposito.php.

De acuerdo al ejemplo anterior de un dao, generar una clase en PHP que contenga 4 métodos, uno para cada elemento del CRUD de la tabla de DEPOSITOS.

C- create – InsertaDeposito

R – read - ObtenDeposito

U – update – actualizaDeposito

D – delete – eliminaDeposito

En el ejemplo siguiente se muestra una clase Fachada. Y Contiene un método que llama al método de la clase DAO y regresa el valor obtenido de ésa función.

OJO: los nombres de las funciones de DAO y FAC pueden llamarse igual o diferente.

```php
<?php

/*
AUTOR: Gilberto Almanza Maldonado
FECHA: 11 de Noviembre de 2020, 15:52 hrs
*/

class facUsuarios{
    public function regresaUsuarios($idNegocio){
        $usuariosArray = array();
        $usuarios      = new daoUsuarios();
        $usuariosArray = $usuarios->regresaUsuarios($idNegocio);
        return $usuariosArray;
    }


}
```

Realizar una script en PHP para la clase Fachada de la tabla de DEPOSITOS. El nombre del método en la fachada para cada método que se tiene en el script DAO deberá ser diferente.