Trabajo Práctico N° 3

En la tercera entrega del TP realizaremos la programación de todos los scripts de los objetos necesarios para poder dar funcionalidad al sistema.

Objetivo: Comprender y profundizar el funcionamiento de la Programación Orientada a Objetos utilizando el lenguaje de programación PHP.

Introducción: En nuestra actualidad vemos que el desarrollo de Internet ha sido inminente y con ello las aplicaciones Web, por lo tanto, se hace indispensable el uso de un lenguaje que permita desarrollar aplicaciones Web como PHP, entre otros. Teniendo en cuenta la situación anterior es que veremos la como desarrollar aplicaciones Web que se ejecutarán del lado del servidor utilizando uno de los mejores lenguajes del ambiente del Software Libre.

Puntos a realizar en la entrega:

1. Crear las clases **TipoDocumento**, **Provincia**, **Sexo** y **Contacto**.
2. Crear la clase **Usuario**.
3. Crear la clase **Persona**.
4. Actuializar los scripts **Paso1.php**, **Paso2.php**, **Paso3.php** y **Finalizar.php**

Desarrollo

1. Crear las clases TipoDocumento, Provincia, Sexo y Contacto.

Cree las siguientes clases con sus atributos y métodos detallados a continuación.

TipoDocumento

- Atributos privados:
  - **_idTipoDocumento**.
  - **_descripcion**.
- Métodos:
  - **Constructor**: debe recibir como parámetro el id y la descripción, y asignarlo al atributo que corresponda.
  - Métodos **getter** para ambos atributos.

Sexo

- Atributos privados:
  - **_idSexo**.
  - **_descripcion**.
- Métodos:
  - **Constructor**: debe recibir como parámetro el id y la descripción, y asignarlo al atributo que corresponda.
  - Métodos **getter** para ambos atributos.

Provincia

- Atributos privados:
  - **_idProvincia**.
  - **_descripcion**.
- Métodos:
  - **Constructor**: debe recibir como parámetro el id y la descripción, y asignarlo al atributo que corresponda.
  - Métodos **getter** para ambos atributos.

Contacto

- Constantes de clases:
  - **TIPO_TELEFONO** con valor 1.
  - **TIPO_EMAIL** con valor 2.
- Atributos privados:
  - **_tipo**.
  - **_valor**.
- Métodos:
  - **Constructor**: debe recibir como parámetro el tipo y el valor, y asignarlo al atributo que corresponda.
  - **validar**: debe verificar que el atributo **_tipo** sea alguno de la constante definida en la clase, y que el contenido del atributo **_valor** cumpla con las siguientes condiciones. Si **_tipo** es igual a **TIPO_TELEFONO**, **_valor** deberá estar compuesto por al menos 10 dígitos separados por un guión. Y si **_tipo** es igual a **TIPO_EMAIL**, **_valor** deberá

contener un símbolo de @. Si estas condiciones se cumplen se retornará el valor de verdadero, de lo contrario se retornara falso.

- Métodos **getter** para ambos atributos.

2. Crear la clase Usuario.

Cree una clase Usuario que cumpla con la siguiente funcionalidad.

- Atributos privados:
  - **_nombre**.
  - **_contrasenia**.
- Metodos:
  - **Constructor**: debe recibir como parámetro el nombre de usuario y la contraseña, y asignarlo al atributo que corresponda.
  - **validarContrasenia**: debe verificar que la contraseña sea de al menos 6 caracteres, y contenga al menos un número y una letra retornando verdadero o falso dependiendo si se cumplen las condiciones.
  - **getContraseniaEnmascarada**: deberá devolver un string reemplazando cada caracter de la contraseña con un símbolo de **"*"**.
  - Métodos **getter** para ambos atributos.

3. Crear la clase Persona.

Cree una clase Persona que cumpla con la siguiente funcionalidad.

- Atributos privados:
  - **_apellido** tipo String.
  - **_nombre** tipo String.
  - **_numeroDocumento** tipo int.
  - **_tipoDocumento** tipo TipoDocumento.
  - **_sexo** tipo Sexo.
  - **_usuario** tipo Usuario.
  - **_nacionalidad** tipo String.
  - **_email** tipo Contacto.
  - **_telefono** tipo Contacto.
  - **_celular** tipo Contacto.
  - **_domicilio** tipo String.
  - **_provincia** tipo Provincia.
  - **_localidad** tipo String.

- Metodos:
  - **Constructor**: no recibirá ningún parámetro pero deberá inicializar los atributos **_tipoDocumento**, **_sexo**, **_usuario**, **_email**, **_telefono**, **_celular**, y **_provincia** con los objetos de sus respectivos tipos.
  - Métodos **getter** y **setter** para sus atributos y teniendo en cuenta pasar **objetos como parámetro** para los **setter** de los atributos que correspondan.

4. Actuializar los scripts Paso1.php, Paso2.php, Paso3.php y Finalizar.php.

Actulizar los scripts de las diferentes etapas de registro de usuario utilizando las distintas clases creadas en los puntos anteriores teniendo en cuenta las siguientes consideraciones.

Paso1.php

- Crear un **array** que contenga 3 objetos del tipo **TipoDocumento** con la información:

ID	Descripción
1	DNI
2	LC
3	LE

- Crear un **array** que contenga 2 objetos del tipo **Sexo** con la información:

ID	Descripción
M	Masculino
F	Femenino

- La información que antes se guardaba en varias claves del array **$_SESSION** ahora deberá guardarse en una sola clave que contenta un objeto del tipo **Persona**.
- Mostrar en cada uno de los componentes del formulario el valor correspondiente con la información del objeto **Persona**.
- Deberá tener en cuenta que la información de las opciones del componente **select** de tipo de documento se generarán con los valores de los objetos creados previamente.
- La misma consideración se deberá tener para los componentes del tipo **radio** para las opciones de sexo.

Paso2.php

- Crear los **arrays** de los tipos **TipoDocumento** y **Sexo** de igual manera que en el script **Paso1.php**, y además crear el siguiente array del tipo **Provinvia**:

ID	Descripción
1	Entre Ríos
2	Santa Fé
3	Córdoba
4	Buenos Aires
5	Catamarca
6	Corrientes

- Restaurar el objeto de tipo **Persona** de la sesión y asignarle los valores recibidos desde el formulario del **Paso1.php** teniendo en cuenta que los valores para las claves **sexo** y **tipo_documento** deben pertenecer a uno de los objetos contenidos en los array con los tipos de documento y sexo, y el valor de la clave **contrasenia** debe cumplir las condiciones que requiere el método **validarContrasenia** de la clase **Usuario**. Si alguna de estas consideraciones no se cumple, se deberá informar al usuario para que vuelva a ingresar la información.
- Si la información ingresada no contiene ningún error, actualizar la variable de sesión con el objeto de tipo **Persona** modificado.
- Muestre en cada uno de los componentes del formulario el valor correspondiente con la información del objeto **Persona**.
- Deberá tener en cuenta que la información de las opciones del componente **select** de provincia se generarán con los valores de los objetos creados previamente.

Paso3.php

- Crear nuevamente el **array** con las provincias como en el **Paso2.php**.
- Restaurar el objeto de tipo **Persona** de la sesión y asignarle los valores recibidos desde el formulario del **Paso1.php** teniendo en cuenta que el valor de la clave **provincia** debe pertenecer a uno de los objetos contenido en el array creado previamente. También que los valores de las claves **celular**, **telefono**, y **email** deben cumplir con las condiciones del método **validar** de la clase **Contacto**. Si alguna de estas condiciones no se cumple se deberá informar al usuario para que vuelva a completar la información.
- Si la información ingresada no contiene ningún error, actualizar la variable de sesión con el objeto de tipo **Persona** modificado.
- Muestre la información del objeto de tipo **Persona** dentro de las etiquetas correspondientes, y recuerde que no se deberá mostrar la contraseña ingresada, sino la respuesta del método **getContraseniaEnmascarada** de la clase **Usuario**.

Entrega

La cuarta entrega deberá contener el proyecto con todos los scripts modificados, y deberá ser cargado a la plataforma como un único archivo comprimido en formato *.rar o *.tar.gz.