Clase Persona en PHP

Aprendiz:

Nicolas David Peña Gómez

Instructor:

Ing. Néstor Montaño

Tecnólogo Análisis y Desarrollo de Software

Ficha: 3064241

SENA Centro de Diseño y Metrología

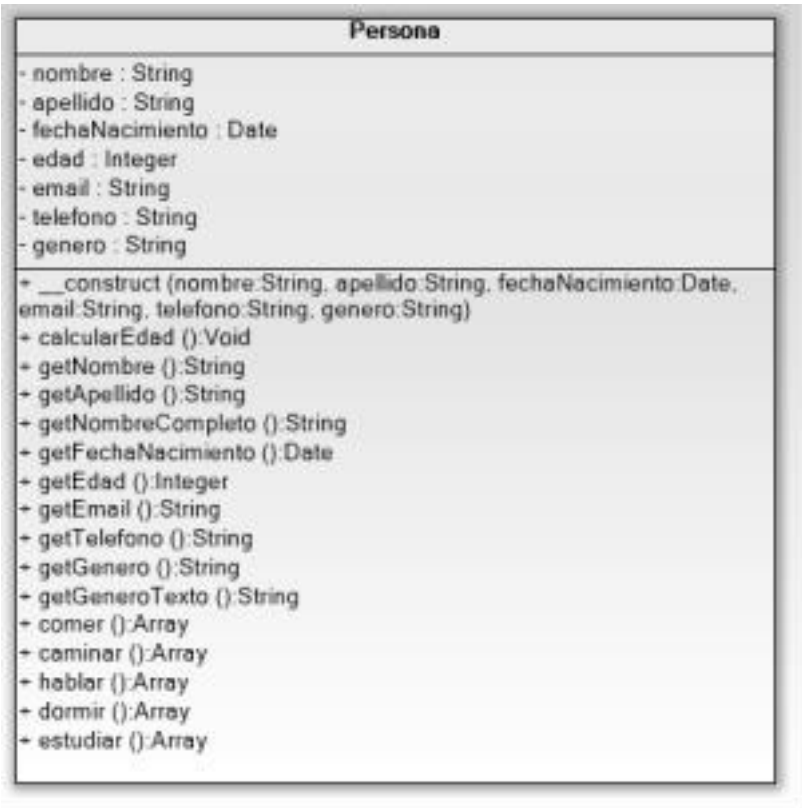
Descripción general

Este proyecto tiene como propósito desarrollar una **aplicación web en PHP** basada en la clase **Persona**, tomando como referencia el **diagrama UML** que define su estructura, atributos y métodos principales.

El trabajo combina los conceptos de la **Programación Orientada a Objetos (POO)** con el desarrollo web, implementando la creación, manipulación y visualización de datos de personas a través de formularios y archivos JSON para su almacenamiento.

Diagrama UML de la Clase Persona

A continuación, se muestra el diagrama UML utilizado como base para el diseño de la aplicación:



Estructura de la clase

La clase Persona está modelada con los siguientes atributos y métodos:

Atributos:

• nombre: String

• apellido: String

• fechaNacimiento: Date

• edad: Integer

• email: String

• telefono: String

• genero: String

Métodos principales:

• __construct(nombre, apellido, fechaNacimiento, email, telefono, genero) • calcularEdad() — calcula la edad con base en la fecha de nacimiento.

• getNombreCompleto() — retorna el nombre y apellido concatenados. • Métodos *getter* para obtener cada dato individual.

• Métodos de acción (comer (), caminar (), hablar (), dormir (), estudiar ()) que simulan actividades mediante arreglos o mensajes predefinidos.

Descripción del desarrollo

El proyecto fue desarrollado completamente en **PHP** aplicando los principios de **POO**, donde se definió la clase Persona en el archivo persona.php y se implementó la interfaz visual en el archivo index.php.

3

1. **Archivo persona.php:**

Contiene la definición de la clase, su constructor, métodos de acceso, cálculo de edad y las funciones que simulan comportamientos humanos.

2. **Archivo index.php:**

Gestiona el formulario de entrada de datos (nombre, apellido, fecha de nacimiento, correo, teléfono, género) y la visualización de los resultados.

Se emplea HTML, CSS y un poco de JavaScript para mostrar mensajes de confirmación y un diseño limpio.

3. **Archivo personas.json:**

Actúa como almacenamiento de datos. Cada vez que se registra una persona, su información se guarda en formato JSON, permitiendo mantener persistencia sin necesidad de una base de datos.

Repositorio del proyecto

El código fuente completo se encuentra disponible en el siguiente enlace: **NicolasP03/CLASE-NESTOR**

Dentro del repositorio podrás encontrar la carpeta llamada **class_personaphp**, que contiene los tres archivos principales:

- **index.php**
- **persona.php**
- **personas.json**

Cómo poner el proyecto a funcionar

1. Descargar el proyecto

- Dirígete al enlace del repositorio en GitHub (ver apartado anterior).
- Da clic en el botón verde **"Code" → "Download ZIP"**.
- Extrae la carpeta descargada llamada **class_personaphp**.

También puedes clonarlo directamente con:

git clone https://github.com/NicolasP03/CLASE-NESTOR.git

2. Requisitos previos

• Tener instalado **XAMPP** (para ejecutar Apache y PHP).

3. Colocar la carpeta del proyecto

• Copia la carpeta **class_personaphp** dentro del directorio de XAMPP:

C:\xampp\htdocs\

Debe quedar así:

C:\xampp\htdocs\class_personaphp\

4. Iniciar el servidor

• Abre el XAMPP Control Panel.
• Da clic en **Start** sobre el módulo **Apache**.

5. Ejecutar el proyecto

• Abre tu navegador web y escribe:

http://localhost/ class_personaphp /

• Se abrirá la interfaz web del proyecto.
Allí podrás ingresar los datos de una persona y ver la información procesada (edad calculada, datos mostrados y guardados en el archivo **JSON**).

6. Verificar almacenamiento

• Dentro de la carpeta del proyecto, abre el archivo **personas.json**.
Verás los registros guardados con todos los datos ingresados desde el formulario.

Conclusión

Este proyecto demuestra la correcta aplicación de los principios de la **Programación Orientada a Objetos en PHP**, integrando la capa de lógica (clase **Persona**) con una interfaz web funcional y un sistema de almacenamiento en **JSON**.

El trabajo evidencia dominio en el diseño de la clase, creación de métodos, validación de datos, gestión de formularios en PHP y organización del proyecto en un entorno de servidor local con **XAMPP**.