GA7-220501096-AA1-EV02 definir estándares de codificación de acuerdo a plataforma de desarrollo elegida

YULIET FAIZULI PACHON CARO

NÉSTOR FABIAN GUTIERREZ SABOGAL

JORGE MILLER GUTIERREZ OSPINA

SENA

ANALISIS Y DESARROLLO DE SOFTWARE – 2721520

Ivan Leonardo Medina Gomez

Mayo 2024

Informe Técnico:

Estándares de Codificación y Aplicación del Paradigma Orientado a Objetos en Java

Introducción:

El presente informe establece los estándares de codificación y describe la aplicación del paradigma orientado a objetos en el desarrollo de la aplicación de consultorio odontológico en Java. Estos estándares están diseñados para promover la legibilidad, mantenibilidad y coherencia del código, mientras que la aplicación del paradigma orientado a objetos asegura un diseño modular y flexible que facilita el desarrollo del software.

Objetivo:

El objetivo de este informe es proporcionar pautas claras y prácticas para el desarrollo de la aplicación de consultorio odontológico en Java, que permitan un código limpio, estructurado y fácil de mantener. Además, se busca aplicar los principios del paradigma orientado a objetos para crear un diseño de software modular, extensible y eficiente.

Estándares de Codificación:

1. Nomenclatura de Variables:

Las variables deben tener nombres descriptivos que reflejen su propósito y significado.

Utilizar notación camelCase para nombrar variables, por ejemplo: `nombrePaciente`, `fechaNacimiento`.

2. Declaración de Clases:

Los nombres de las clases deben comenzar con una letra mayúscula y seguir la convención CamelCase.

Cada clase debe tener una única responsabilidad y representar una entidad coherente del dominio del problema, como `Paciente`, `Cita`, `Tratamiento`, etc.

3. Declaración de Métodos:

Los nombres de los métodos deben ser descriptivos y expresar la acción que realizan.

Utilizar notación camelCase para nombrar métodos, por ejemplo: `agregarPaciente`, `buscarCita`.

4. Encapsulación:

Los atributos de las clases deben ser privados y accederse a través de métodos de acceso (getters) y métodos de modificación (setters).

Evitar la exposición directa de los atributos públicos para garantizar la cohesión y el control del acceso a los datos.

5. Herencia y Polimorfismo:

Utilizar la herencia cuando exista una relación de especialización entre las clases.

Aplicar el polimorfismo para permitir que los objetos se comporten de diferentes maneras según el contexto.

6. Abstracción e Interfaces:

Definir clases abstractas e interfaces para modelar conceptos generales y comportamientos comunes.

Utilizar interfaces para establecer contratos y definir comportamientos que pueden ser implementados por múltiples clases.

7. Comentarios y Documentación:

Incluir comentarios claros y concisos para explicar el propósito y el funcionamiento de clases y métodos.

Utilizar el formato Javadoc para documentar clases, interfaces y métodos públicos, proporcionando descripciones detalladas y ejemplos de uso.

Al seguir estos estándares de codificación y aplicar los principios del paradigma orientado a objetos en el desarrollo de la aplicación de consultorio odontológico en Java, se asegurará un código limpio, modular y fácil de mantener, lo que facilitará el desarrollo y la evolución del software a lo largo del tiempo.