**SERVICIO NACIONAL DE APRENDIZAJE SENA REGIONAL CAUCA**

**DEFINIR ESTÁNDARES DE CODIFICACIÓN DE ACUERDO A PLATAFORMA**

**DE DESARROLLO ELEGIDA**

**GA7-220501096-AA1-EV02**

**PRESENTA:**

**JOANA ANDREA SILVA MACIAS**

**CC: 1125181139**

**INSTRUCTOR:**

**AREA TECNICA**

**INFORMACIÓNTECNOLOGIA EN ANALISIS Y DESARROLLO DE SOFTWARE**

**FICHA: 2977349**

**BOGOTA 27/05/2025**

**INTRODUCCION**

En la programación orientada a objetos, los objetos se modelan como entidades que tienen un estado, un comportamiento y una identidad. Estos objetos interactúan entre sí mediante el intercambio de mensajes, lo que permite la creación de programas más modulares y reutilizables. Además, la programación orientada a objetos proporciona herramientas para la abstracción, encapsulamiento, herencia y polimorfismo, que son conceptos fundamentales para la creación de programas más complejos. Este paradigma se basa en la idea de que un programa está compuesto por objetos, que son entidades que tienen datos (atributos) y acciones (métodos). La POO facilita la reutilización de código, la organización de programas complejos y la creación de interfaces de usuario. Lenguajes como Java, Python, C++ y C# son ejemplos de lenguajes orientados a objetos.

Java es un lenguaje de programación potente y versátil que ha sido ampliamente adoptado en la industria debido a su portabilidad, seguridad y facilidad de uso. Es una herramienta indispensable para el desarrollo de una amplia gama de aplicaciones, desde pequeñas aplicaciones de escritorio hasta sistemas empresariales complejos y aplicaciones móviles.

**OBJETIVO**

La aplicación del paradigma orientado a objetos tiene como objetivo principal modelar objetos del mundo real en un programa de computadora y permitir su interacción de manera más natural y sencilla. Entre sus objetivos específicos se encuentran la facilitación de la creación de programas más complejos, la mejora de la escalabilidad, la eficiencia en el desarrollo, y la reutilización del código.

**ESTÁNDARES DE CODIFICACIÓN:**

Nombramiento de variables:

Una clase en java se compone por las siguientes declaraciones:

* Paquete
* Comentarios
* Definición de clases
* Atributos
* Constructo

Métodos Java determina en sus estándares de codificación convenciones para nombres de clases, métodos y atributos. Para las clases se define lo siguiente:

1. La primera letra debe ser mayúscula.
2. Utilizar nomenclatura camelCase (por ejemplo: un nombre compuesto por varias palabras: clase "CuentaCredito").
3. Los nombres deben ser sustantivos.

**Ejemplo**

Estructura: public class NombreDeLaClase {}

Donde la palabra **‘public’** indica que el alcance de la clase será público.

La palabra **‘class’** indica que es una clase. Y las llaves {} se usan para agrupar los bloques de código Todos los identificadores Java presentan unas condiciones en su nomenclatura que se resumen en los siguientes aspectos:

* Deben tener nombre claro y conciso Se permite el uso del carácter subrayado (\_), exceptuando la posición final e inicial del identificador.
* No se permiten dos caracteres subrayados consecutivos.

El nombre de la clase, Publicación debe ser un identificador válido en Java y por convención, empieza por letra mayúscula. Una declaración de clase crea un nombre de tipo en Java. De esta forma las referencias a objetos Publicación se pueden declarar con un simple:

**Nombre de la clase:**

Las variables de una clase se llaman campos. La clase Publicación tiene tres campos, idPublicacion que identifica de forma unívoca cada publicación, título que es el título de la publicación y autor que ha firmado la publicación.

La creación de un objeto en Java es equivalente a decir que se está instanciando una clase. Cuando se hace la instancia se comparten los atributos y métodos de esta. Una sola clase puede tener varias instancias.

**Declaración de un método:**

Un método consta de un encabezado y un cuerpo. Para declarar el encabezado de un método, basta con escribir el tipo que retorna, seguido del nombre del método y entre paréntesis la lista de parámetros. Ejemplos: void imprimir(); este método no retorna nada y no tiene parámetros.

**Las Convenciones:**

Son para el nombramiento de clases, métodos y atributos que ayudan a que el código sea más legible y consistente.

* **Nombres de clases:**

Utiliza UpperCamelCase.

Los nombres deben ser sustantivos y comenzar con una letra mayúscula. Ejemplo: MiClase, Persona, CuentaBancaria.

* **Nombres de métodos:**

Utiliza lowerCamelCase.

Los nombres deben ser verbos o frases verbales que describan la acción realizada por el método.

Ejemplo: calcularTotal, obtenerNombre, validarUsuario.

* **Nombres de atributos:**

Utiliza lowerCamelCase.

Los nombres deben ser descriptivos y representar el propósito o la característica del atributo.

Ejemplo: nombre, edad, saldoCuenta.

* **Nombres de variables locales:**

Nombres de variables locales:

Utiliza lowerCamelCase.

Los nombres deben ser concisos pero descriptivos.

Ejemplo: contador, resultado, nombreUsuario.

* **Nombres de constantes:**

• Nombres de constantes:

Utiliza MAYÚSCULAS\_CON\_GUIONES\_BAJOS.

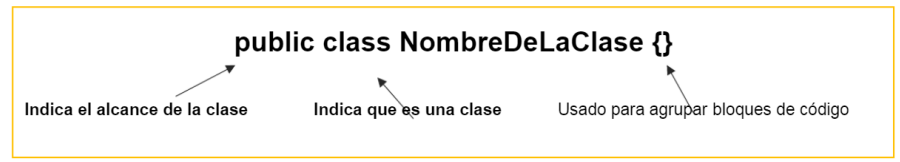
Los nombres deben ser descriptivos y utilizar palabras completas.

Ejemplo: VALOR\_MAXIMO, PI, TASA\_INTERES.

* **Nombres de paquetes:**

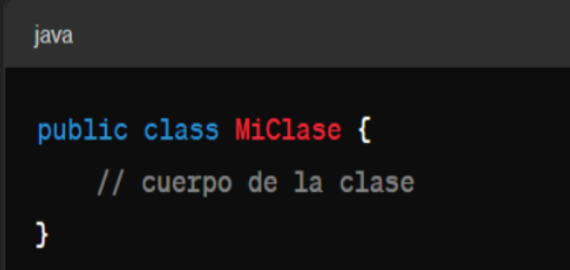
Utiliza minúsculas. Los nombres deben ser únicos y reflejar la estructura jerárquica del proyecto.

Ejemplo: com. miempresa. proyecto, org. biblioteca. utilidades. Estructura

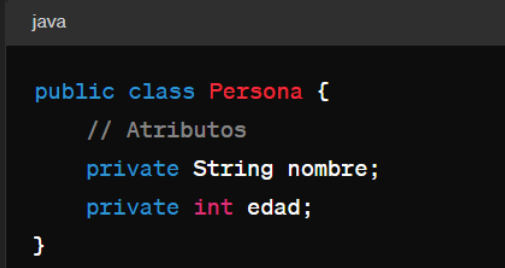


* En Java, la estructura de una clase sigue un patrón definido que incluye diferentes elementos. Aquí tienes una descripción de los componentes comunes de una clase en Java:

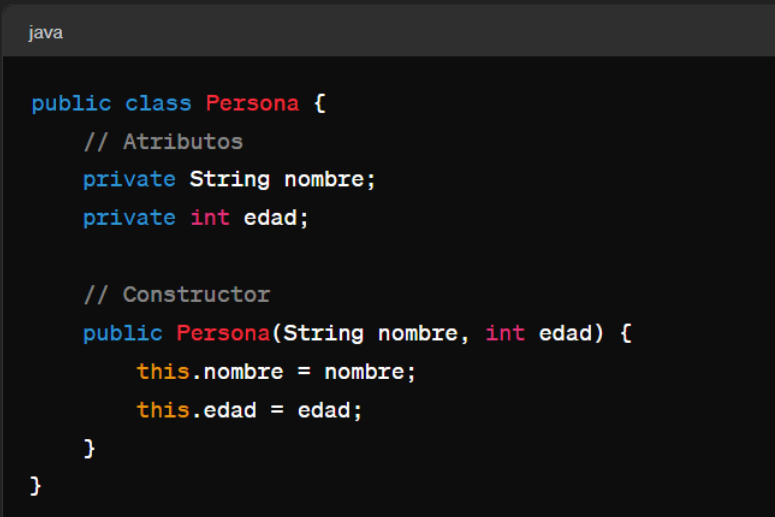
1. **Declaración de clase:** Comienza con la palabra clave class, seguida del nombre de la clase y las llaves {} que contienen el cuerpo de la clase. Por ejemplo:



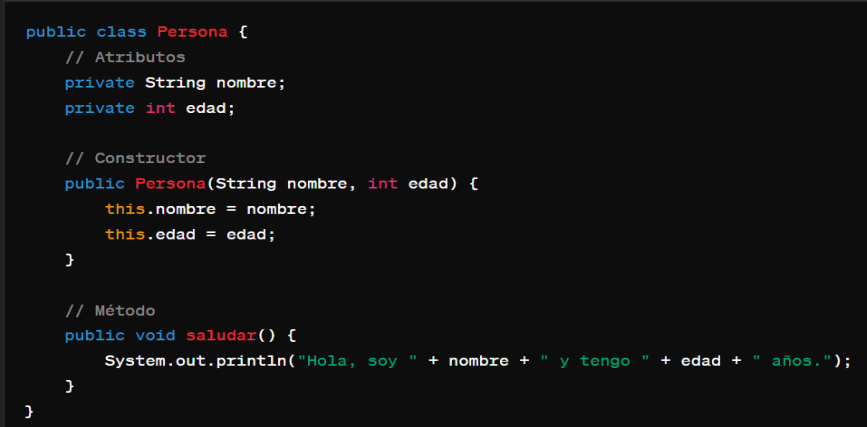
1. **Atributos:** (variables de instancia): Son variables que representan el estado de un objeto y están asociadas a una instancia específica de la clase. Se definen dentro de la clase, pero fuera de cualquier método. Por ejemplo:



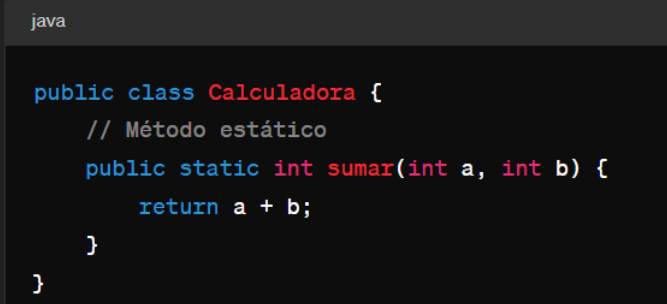
1. Constructores: Son métodos especiales que se utilizan para inicializar objetos de la clase. Tienen el mismo nombre que la clase y no tienen tipo de retorno. Pueden tener parámetros para pasar valores iniciales a los atributos de la clase. Por ejemplo:



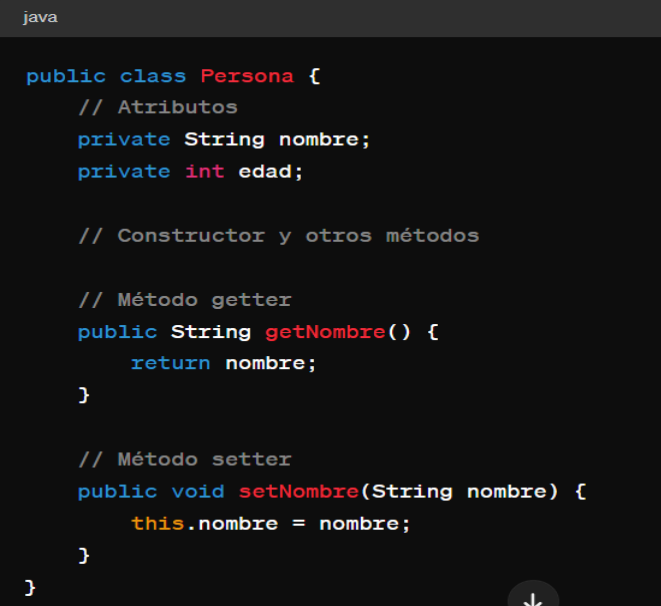
1. **Métodos:** Son funciones que definen el comportamiento de los objetos de la clase. Se definen dentro de la clase y pueden acceder y manipular los atributos de la clase. Por ejemplo:



1. Métodos estáticos: Son métodos que se asocian con la clase en lugar de con instancias individuales de la clase. Se definen con la palabra clave static. Por ejemplo:



1. **Métodos getter y setter:** Son métodos utilizados para acceder y modificar los valores de los atributos de la clase (métodos de acceso). Los getters se utilizan para obtener el valor de un atributo, mientras que los setters se utilizan para establecer el valor de un atributo. Por ejemplo:



**CONCLUSION**

Seguir estándares de codificación en Java no solo beneficia al equipo de desarrollo al mejorar la productividad y la calidad del código, sino que también contribuye a la creación de software más robusto y satisfactorio para los usuarios finales.

La programación orientada a objetos es un paradigma de programación poderoso que permite modelar objetos del mundo real en un programa de computadora y facilita la creación de programas más complejos y escalables. En la actualidad, su aplicación se ha vuelto cada vez más popular debido a sus ventajas en la creación de programas más modulares, reutilizables y mantenibles.

Referencias

https://www.youtube.com/watch?v=vLIAl7q\_i-I&t=11s

https://www.youtube.com/watch?v=e8gaffa3Ca8&t=3413s

**REFERENCIAS**

* Java en Castellano. (2021, Dic 01). Servlets y JSP.

<https://programacion.net/articulo/servlets_y_jsp_82>

* Maida, E. G., & Pacienzia, J. (2015). Metodologías de desarrollo de software.
* Quintas, A. F. (2000). Java 2: manual de usuario y tutorial. Grupo Editorial RA-MA.