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

Aprendiz: José Carlos Pinzón Ferrer

Servicio Nacional de Aprendizaje Tecnólogo en Análisis y Desarrollo de Software

Instructor: Cristian Camilo Arias

Bogotá, Colombia 18 de febrero de 2024

Introducción

La programación orientada a objetos (POO) ofrece una forma eficiente y estructurada de diseñar sistemas de software, permitiendo a los programadores modelar problemas de manera más fiel a la realidad. Este documento explora las características fundamentales de la programación orientada a objetos, así como los estándares de codificación asociados y ejemplos relevantes de su aplicación en diferentes lenguajes.

Objetivo

El propósito de este informe es ofrecer una perspectiva amplia de la programación orientada a objetos, incluyendo sus conceptos esenciales, directrices de codificación y ejemplos aplicados. Se examinó las definiciones básicas de objetos y clases, así como los principios fundamentales como la encapsulación, abstracción, herencia y polimorfismo que constituyen la base de este enfoque de programación.

¿Qué es POO?

La programación orientada a objetos se ha convertido en uno de los enfoques más populares y utilizados para el desarrollo de software, y se usa en una amplia variedad de lenguajes de programación, desde Java y C++ hasta Python y Ruby.

Entonces, podemos definir que la orientación a objetos en la programación es un paradigma donde se estructura el diseño del software alrededor de entidades llamadas objetos, en contraposición al enfoque basado en funciones y lógica. Este enfoque pone énfasis en los objetos que deben ser manipulados por los programadores, en lugar de focalizarse en la lógica requerida para dicha manipulación. Cada objeto posee atributos y comportamientos particulares, definiéndose esencialmente como una entidad que encapsula datos y funcionalidades específicas.

El paradigma orientado a objetos nos permite modelar un problema de una forma amena, que reproduce con mayor fidelidad los mecanismos del mundo real. Todos los elementos que componen el Universo tienen características y comportamiento propios, y se relacionan con otros elementos de múltiples formas; por lo tanto, pueden ser fácilmente representados como objetos. Crear un conjunto de objetos que refleje adecuadamente el problema real, y que sepan organizarse entre sí para resolverlo, depende en buena medida de la imaginación del programador.

Definiciones

Objeto

Objeto es cualquier cosa tangible o intangible que se pueda imaginar definida mediante sus atributos y las operaciones que permiten modificar dichos atributos. En la POO se dice que un objeto: "integra datos y algoritmos". La estructurada, variables y funciones están separadas.

Cada objeto es responsable de inicializarse y destruirse en forma correcta, además consta de:

- Tiempo de vida: La duración de un objeto en un programa siempre está limitada en el tiempo
  La mayoría de los objetos sólo existen durante una parte de la ejecución del programa
  Los objetos son creados mediante un mecanismo denominado instanciación
  Los objetos dejan de existir cuando son destruidos
- Estado: Queda definido por sus atributos
- Comportamiento: Con él se definen las propiedades del objeto, y elnestado en que se encuentra en un momento ndeterminado de su existencia.

Clase

Una clase es una plantilla que permite definir un conjunto de objetos. Son abstracciones que representan a un conjunto de objetos con un:

- Comportamiento
- Interfaz común
- Es la implementación de un tipo de dato (considerando los objetos como instancias de las clases)
- Permiten definir y representar colecciones de objetos
- Proveen un modelo para la creación de objetos.

Atributos y variables de una Clase: Representan el estado de un objeto. Una variable es un espacio físico de memoria donde un programa puede almacenar un dato para su posterior utilización. Los atributos de una clase son definidos según esta sintaxis:

[modifVisibilidad] [modifAtributo] tipo nombreVariable [=valorInicial] .

Los nombres de las variables empiezan con una letra minúscula. Los nombres de las clases empiezan con una letra mayúscula el nombre de variable no debe tener el mismo nombre que otras variables.

Métodos. Representan el comportamiento de un objeto (funciones miembros). Un método es una abstracción de una operación que puede hacer o realizarse con un objeto. Una clase puede declarar cualquier número de métodos que lleven a cabo operaciones diversas con los objetos. Los métodos permiten manipular los datos definidos por la clase y, en muchos casos, brindan acceso a esos datos.

Tipos de métodos. Hay varios tipos de métodos que son comunes a la mayoría de las

clases:

- **Constructores.** Un constructor es un método que tiene el mismo nombre que la clase. Se ejecuta cuando se crea un objeto de una clase. Por lo tanto, un constructor contiene instrucciones para inicializar las variables de un objeto.
- **Destructores.** Un destructor es un m´etodo que se utiliza para destruir un objeto.
- No todos los lenguajes orientados a objetos poseen destructores.
- **Accesores**. Un accesor es un método que devuelve el valor de un atributo privado de otro objeto. Así es cómo los objetos externos pueden acceder a los datos encapsulados.
- **Mutadores.** Un mutador es un método que almacena un nuevo valor en un atributo. De este modo es cómo objetos externos pueden modificar los datos encapsulados.

Entre las características de la POO podemos indicar los siguientes conceptos:

Encapsulación. La implementación y el estado de cada objeto se mantienen de forma privada dentro de un límite definido o clase. Otros objetos no tienen acceso a esta clase o la autoridad para realizar cambios, pero pueden llamar a una lista de funciones o métodos públicos. Esta característica de ocultación de datos proporciona una mayor seguridad al programa y evita la corrupción de datos no intencionada.

Abstracción. Los objetos solo revelan mecanismos internos que son relevantes para el uso de otros objetos, ocultando cualquier código de implementación innecesario. Este concepto ayuda a los desarrolladores a realizar cambios y adiciones más fácilmente a lo largo del tiempo.

Herencia. Se pueden asignar relaciones y subclases entre objetos, lo que permite a los desarrolladores reutilizar una lógica común sin dejar de mantener una jerarquía única. Esta propiedad de OOP obliga a un análisis de datos más completo, reduce el tiempo de desarrollo y asegura un mayor nivel de precisión.

La herencia facilita la reutilización de código y la abstracción.

```
(define persona%

(class object%

(init-field nombre)

(field (amigos '()))

(define/public (di-hola)

(printf "hola, me llamo ~a~%" nombre))

(define/public (get-nombre)

nombre)

(define/public (es-amigo? otro)

(if (memq otro amigos) ;memq comprueba si otro está en

amigos usando la igualdad eq?
```

```
#t
#f))
(define/public (añade-amigo otro)
(if (not (es-amigo? otro)) ; tambien es posible llamar a los
métodos directamente
(begin
(set! amigos (cons otro amigos))
(send otro añade-amigo this))))
(define/public (saludan-amigos)
(for-each
(lambda (f) (send f di-hola))
amigos))
(super-new)))
(define frodo (new persona% (nombre "Frodo")))
(define gandalf (new persona% (nombre "Gandalf")))
(send frodo añade-amigo gandalf)
(send frodo saludan-amigos)
(send gandalf saludan-amigos)
```

Polimorfismo. Los objetos pueden adoptar más de una forma según el contexto. El programa determinará qué significado o uso es necesario para cada ejecución de ese objeto, reduciendo la necesidad de duplicar código.

Estándares de Codificación:

Nombramiento de Variables: Utilizar nombres descriptivos que reflejen el propósito y el contenido de la variable. Seguir convenciones como camelCase o snake_case según las normas del lenguaje.

Declaración de Clases: Nombrar las clases utilizando sustantivos que representen adecuadamente la entidad que están modelando. Utilizar mayúsculas para la primera letra de cada palabra en el nombre de la clase (CamelCase).

Declaración de Métodos: Utilizar verbos que describan la acción que realiza el método. Seguir convenciones de nombramiento similares a las variables, pero con la primera letra en minúscula (camelCase).

Organización del Código: Mantener una estructura clara y coherente en el código, agrupando variables y métodos relacionados dentro de las clases correspondientes. Utilizar comentarios para explicar el propósito de las secciones de código o aclarar su funcionamiento cuando sea necesario.

Documentación: Incluir documentación en forma de comentarios en el código para explicar el propósito de las clases, métodos y variables, así como sus parámetros y valores de retorno.

Ejemplos de lenguajes de programación orientada a objetos

Simula

Aunque ya hoy no se utilice, Simula es merecedor de este primer lugar debido a que ha sido el primer lenguaje de programación orientada a objetos que incluyó el concepto de clase.

Este POO desarrollado por Ole Johan Dahl y Kristen Nygaard en 1962 influyó en casi todos los lenguajes de programación moderno. Esto es gracias a muchas características interesantes como el polimorfismo, los objetos y las instancias, entre otras.

Java

Java es uno de los lenguajes de programación orientados a objetos más importantes de la actualidad. Además de un lenguaje de programación, Java es una plataforma. Es decir que se requiere instalar una máquina virtual Java para poder correr las aplicaciones creadas con el lenguaje.

Fue lanzada al mercado por Sun Microsystems en 1995, y al día de hoy tiene una relevancia más que importante. Este lenguaje de programación orientado a objetos está presente en muchas implementaciones, tanto empotrado en dispositivos como en aplicaciones para celulares y computadoras.

ADA

Ada es un lenguaje de programación orientado a objetos estático de propósito general y fuertemente tipado. Fue diseñado en 1983 por Jean Ichbiah de CII Honeywell Bull por encargo del Departamento de Defensa de los Estados Unidos. Fue nombrado de este modo debido a Augusta Ada Byron y está inspirado fuertemente por Pascal.

C++

C++ es un lenguaje de programación orientado a objetos que fue desarrollado por Bjarne Stroustrup en 1979. El propósito de su creación fue mejorar algunos aspectos del lenguaje C para permitir la creación y manipulación de objetos. Más tarde sus desarrolladores le agregaron opciones para la programación genérica, que se sumaron a

sus capacidades de lenguaje de programación estructurada y programación orientada a objetos. Es por ello que C++ es considerado como un lenguaje híbrido y multiparadigma.

C#

Podría considerarse e a “C#” como un lenguaje de programación multiparadigma. Este fue desarrollado por Microsoft con el objetivo de completar su plataforma NET. Cabe destacar que la sintaxis de este lenguaje proviene de C++, y a su vez de C. Para el desarrollo de software C# utiliza el modelo de objetos provisto por NET, plataforma similar a Java.

Ruby

Ruby se comenzó a desarrollar en 1993 y fue presentado en 1995. Su creador es Yukihiro Matsumoto. Se trata de un lenguaje de programación interpretado, reflexivo y orientado a objetos. Además, se distribuye bajo una licencia de software libre. Podría decirse que Ruby es una cruza entre dos lenguajes de programación diferentes. Esta afirmación es debido a que Ruby combina una sintaxis inspirada en Python y Perl, con características de programación orientada a objetos muy parecida al lenguaje Smalltalk.

Python

Otro de los lenguajes de programación orientados a objetos más importantes de la actualidad es Python. Básicamente, Python es un lenguaje de programación interpretado de alto nivel y multipropósito.

PHP

PHP es un lenguaje de código abierto muy popular en la actualidad. Su nombre es el acrónimo recursivo de PHP: Hypertext Preprocessor y es un lenguaje de programación creado Rasmus Lerdorf en 1994. Podría considerarse a PHP como un lenguaje de programación multipropósito, pero con los años se ha adaptado y usado ampliamente en el desarrollo web, ya que puede ser incrustado en HTML.

PowerBuilder

PowerBuilder fue desarrollado como un entorno para crear aplicaciones de gestión de bases de datos. En este sentido, es capaz de trabajar con las bases de datos más populares en estos ámbitos como MS SQL Server o MySQL.Sin embargo, con este lenguaje de programación orientado a objetos se pueden desarrollar otros tipos de aplicaciones cliente/servidor o distribuidas. Otra característica de PowerBuilder es que para programar en este lenguaje hay que hacerlo en un lenguaje propio, llamado Powerscript.

Visual Basic.NET

Visual Basic. NET es un lenguaje de programación orientado a objetos desarrollada con el propósito de crear aplicaciones para la web. Es considerado como el sucesor de Visual Basic, sin embargo, los dos lenguajes son muy distintos entre sí, con lo cual es imposible conseguir retrocompatibilidad, es decir poder usar los proyectos de Visual Basic NET en Visual Basic.

Pantalla de Visual Basic

Al ser parte integrante de los lenguajes de programación que incluye Microsoft Visual Studio, es capaz de aprovechar todas las ventajas que ofrece el entorno NET. Sin embargo, también pude llegar a usarse en el IDE libre SharpDevelop.

Conclusiones

La programación orientada a objetos ofrece un enfoque poderoso y flexible para el desarrollo de software, permitiendo a los desarrolladores crear sistemas complejos de manera estructurada y mantenible. Al centrarse en entidades llamadas objetos, este paradigma facilita la representación de problemas del mundo real y fomenta la reutilización de código a través de conceptos como encapsulación, abstracción, herencia y polimorfismo. Con una comprensión clara de los principios y prácticas asociados con la programación orientada a objetos, como desarrolladores podemos aprovechar al máximo este enfoque en sus proyectos y contribuir al avance continuo de la industria del software.

B**eneficios de Programación Orientada a Objetos**

- Reutilización del código.
- Convierte cosas complejas en estructuras simples reproducibles.
- Evita la duplicación de código.
- Permite trabajar en equipo gracias al encapsulamiento ya que minimiza la posibilidad de duplicar funciones cuando varias personas trabajan sobre un mismo objeto al mismo tiempo.
- Al estar la clase bien estructurada permite la corrección de errores en varios lugares del código.
- Protege la información a través de la encapsulación, ya que solo se puede acceder a los datos del objeto a través de propiedades y métodos privados.
- La abstracción nos permite construir sistemas más complejos y de una forma más sencilla y organizada.

Bibliografía

https://www.computerweekly.com/es/definicion/Programacion-orientada-a-objetos-OOP

https://www.tecnologia-informatica.com/lenguajes-de-programacion-orientada-a-objetos/

http://aisii.azc.uam.mx/mcbc/Cursos/POO/Sesion3.pdf

https://profile.es/blog/que-es-la-programacion-orientada-a-objetos/

https://www3.uji.es/~mmarques/e16/teoria/cap2.pdf