PROGRAMACIÓN II MARKETING

LECTURAS CUARTA SESION

https://www.youtube.com/watch?v=-6BYa_x_QA0

https://laraveles.com/series/poo/la-abstraccion-programacion-orientada-objetos/

¿Qué es la Abstracción en Programación Orientada a Objetos?

Podemos definir la abstracción como las características específicas de un objeto, aquellas que lo distinguen de los demás tipos y que logran definir límites conceptuales respecto a quien está haciendo dicha abstracción .

En otras palabras, es conseguir describir un objeto con propiedades y métodos principales sin pensar en detalle. La abstracción separa el comportamiento específico de un objeto, a esta división que se realiza se le conoce como la barrera de abstracción.

Para conseguir llegar a la barrera de abstracción muchas veces se debe realizar el proceso en el cual la interfaz de un objeto muestre el comportamiento específico y nada más, a este proceso le denomina el proceso de mínimo compromiso.

Una Interfaz de objeto permite crear código con el cuál se específica que métodos serán implementados por una clase sin necesidad de definir que harán estos métodos.

Ejemplo de una Interfaz con PHP

```
interface Animal

{

   public function eat($food);

   public function name();

}
```

Ejemplo de una interfaz con C#

```
interface Animal

{

   string eat(string food);

   string name();

}
```

Como se puede observar en ambos ejemplos se definen dos métodos eat y name ambos únicamente están definidos pero no tienen implementación, ya que la implementación deberá realizarla toda aquella clase que herede de ella.

En algunos casos existe también el conocido principio de mínima sorpresa, en el cuál una abstracción obtiene el comportamiento completo de un objeto y por ende no ofrece sorpresas o efectos laterales que lleguen más allá de la abstracción.

Algunas de las abstracciones que existen son:

Abstracción de Entidades: Es un objeto que representa un modelo útil de una entidad que se desea.

Abstracción de Acciones: Un objeto que representa un conjunto de operaciones y todas ellas desempeñan funciones del mismo tipo.

Abstracción de Máquinas virtuales: Un objeto que agrupa operaciones, todas ellas virtuales, utilizadas por algún nivel superior de control u operaciones (entre ellos se puede mencionar hablar de un circuito).

Abstracción de coincidencia: Un objeto que almacena un conjunto de operaciones que no tienen relación entre sí.

Como entenderás la interfaz permite generar una clase modelo que tenga de manera abstracta la vista exterior de un objeto, dicha interfaz establece todas las propiedades y el comportamiento de un objeto, las interfaces ayudan a limitar la responsabilidad de un objeto es decir le indica al objeto del comportamiento que se le hace responsable.

Cualquier duda o sugerencia estamos atentos, recordando que estamos en una serie de artículos cortos donde cada tema es tratado en un artículo distinto.

OBJETO

En el paradigma de programación orientada a objetos (POO, o bien OOP en inglés), un objeto es una unidad dentro de un programa de computadores que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado instanciando una clase, como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación de otros objetos, como ocurre en la programación basada en prototipos.

Estos objetos interactúan unos con otros, en contraposición a la visión tradicional en la cual un programa es una colección de subrutinas (funciones o procedimientos), o simplemente una lista de instrucciones para el computador. Cada objeto es capaz de recibir mensajes, procesar datos y enviar mensajes a otros objetos de manera similar a un servicio.

En el mundo de la programación orientada a objetos (POO), un objeto es el resultado de la instanciación de una clase.1 Una clase es el anteproyecto que ofrece la funcionalidad en ella definida, pero ésta queda implementada sólo al crear una instancia de la clase, en la forma de un objeto. Por ejemplo: dado un plano para construir sillas (una clase de nombre clase_silla), entonces una silla concreta, en la que podemos sentarnos, construida a partir de este plano, sería un objeto de clase_silla. Es posible crear (construir) múltiples objetos (sillas) utilizando la definición de la clase (plano) anterior. Los conceptos de clase y objetos son análogos a los de tipo de datos y variable; es decir, definida una clase podemos crear objetos de esa clase, igual que disponiendo de un determinado tipo de dato (por ejemplo el tipo entero), podemos definir variables de dicho tipo:

int a,b;

Definición de objeto[editar]

Un objeto en POO representa alguna entidad de la vida real, es decir, alguno de los objetos que pertenecen al negocio con que estamos trabajando o al problema con el que nos estamos enfrentando, y con los que podemos interactuar. A través del estudio de ellos se adquiere el conocimiento necesario para, mediante la abstracción y la generalización, agruparlos según sus características en conjuntos. Estos conjuntos determinan las clases de objetos con las que estamos trabajando. Primero existen los objetos; luego aparecen las clases en función de la solución que estemos buscando. Ésta es la forma más común de adquirir conocimiento aunque no es la única. En ocasiones, cuando el observador es un experto del negocio (o del problema), el proceso puede ser a la inversa y comenzar el análisis en una base teórica abstracta, sustentada por el conocimiento previo que da lugar primeramente a clases de objetos que satisfagan las necesidades de la solución.

Estos conceptos son parte de la base teórica de la idea de objeto y clase utilizados en la POO. Los objetos tienen características fundamentales que nos permiten conocerlos mediante la observación, identificación y el estudio posterior de su comportamiento; estas características son:

Identidad

Comportamiento

Estado

En las ramas de las ciencias de la computación más estrictamente matemáticas, el término objeto es usado en sentido puramente matemático para referirse a cualquier "cosa". Esta interpretación resulta útil para discutir sobre teorías abstractas, pero no es suficientemente concreta para servir como definición de un tipo primitivo en discusiones de ramas más específicas, como en la programación, que está más cerca de cálculos reales y el procesamiento de información.

Identidad

La identificación es la propiedad que permite diferenciar a un objeto y distinguirse de otros. Generalmente esta propiedad es tal, que da nombre al objeto. Tomemos por ejemplo el "verde" como un objeto concreto de una clase color; la propiedad que da identidad única a este objeto es precisamente su "color" verde. Tanto es así que para nosotros no tiene sentido usar otro nombre para el objeto que no sea el valor de la propiedad que lo identifica.

En programación la identidad de todos los objetos sirve para comparar si dos objetos son iguales o no. No es raro encontrar que en muchos lenguajes de programación la identidad de un objeto esté determinada por la dirección de memoria de la computadora en la que se encuentra el objeto, pero este comportamiento puede ser variado redefiniendo la identidad del objeto a otra propiedad.

Comportamiento

El comportamiento de un objeto está directamente relacionado con su funcionalidad y determina las operaciones que este puede realizar o a las que puede responder ante mensajes enviados por otros objetos. La funcionalidad de un objeto está determinada, primariamente, por su responsabilidad. Una de las ventajas fundamentales de la POO es la reusabilidad del código; un objeto es más fácil de reutilizarse en tanto su responsabilidad sea mejor definida y más concreta.

Una tarea fundamental a la hora de diseñar una aplicación informática es definir el comportamiento que tendrán los objetos de las clases involucradas en la aplicación, asociando la funcionalidad requerida por la aplicación a las clases adecuadas.

Estado

El estado de un objeto se refiere al conjunto de atributos y sus valores en un instante de tiempo dado. El comportamiento de un objeto puede modificar el estado de este. Cuando una operación de un objeto modifica su estado se dice que esta tiene "efecto colateral".

Esto tiene especial importancia en aplicaciones que crean varios hilos de ejecución. Si un objeto es compartido por varios hilos y en el transcurso de sus operaciones estas modifican el estado del objeto, es posible que se deriven errores del hecho de que alguno de los hilos asuma que el estado del objeto no cambiará