Introducción a la programación orientada a objetos

Javier Luis Fernández

Administración de sistemas gestores de bases de datos

2º ASIR

¿QUE ES LA PROGRAMACIÓN ORIENTADA A OBJETOS?

La programación Orientada a objetos se define como un paradigma de la programación, una manera de programar específica, donde se organiza el código en unidades denominadas clases, de las cuales se crean objetos que se relacionan entre sí para conseguir los objetivos de las aplicaciones.

Podemos entender la programación Orientada a objetos (POO) como una forma especial de programar, más cercana a como expresaríamos las cosas en la vida real que otros tipos de programación, que permite diseñar mejor las aplicaciones, llegando a mayores cotas de complejidad, sin que el código se vuelva inmanejable.

- **CLASES**
  Las clases son una parte fundamental de la programación orientada a objetos.Usas clases para representar cualquier entidad que tenga algunas propiedades y funciones que puedan actuar sobre propiedades determinadas. TypeScript le brinda control total sobre las propiedades y funciones que son accesibles dentro y fuera de su propia clase contenedora

CARACTERISTICAS

- **ABSTRACCIÓN**
  Hay una sintaxis especial en TypeScript para indicar que una clase es abstracta. Dentro de la clase abstracta definimos un método abstracto llamado operar, esto obliga a todas las clases que heredan de Operación implementar el algoritmo de dicho método, sino se genera un error en tiempo de compilación .

- **MODULARIDAD**
  La modularización es el proceso por el cual seleccionamos y agrupamos instrucciones de programación que cumplen una función específica.

- **ENCAPSULAMIENTO**
  En programación orientada a objetos, se denomina encapsulamiento al ocultamiento del estado, es decir, de los datos miembro de un objeto de manera que solo se pueda cambiar mediante las operaciones definidas para ese objeto.

- **JERARQUIA**
  La herencia en Typescript es clásica ya que se basa en un sistema jerárquico, en lugar de basarse en una cadena de objetos anidados como es el caso de Javascript. Este tipo de herencia es más compleja que la de Javascript, incorpora nuevos conceptos y mecanismos de herencia que hacen más potente pero a su vez más complejo al lenguaje. Conceptos tales como interfaces, atributos públicos, atributos privados, método públicos, métodos privados, constructores ....

- **POLIMORFISMO**
  Es una propiedad con la que podemos definir un método en la clase principal y permite que las subclases que lo heredan implementen cada subclase con un rendimiento diferente .

MODIFICADORES DE ACCESO

El modificador public es el que viene por defecto en Typescript; cuando no indicas ningún modificador, el lenguaje pone por defecto este modificador. Lo que hace, es hacer visibles para todo el mundo el atributo que lo precede.

El modificador private, por otro lado, es lo opuesto a public. Este modificador restringe la visibilidad de los atributos de la clase sólo a esta clase. Dicho de otra manera, nadie más que la clase tiene acceso a estos atributos, tanto para leer como para modificarlos.

El modificado protected hace visible atributos entre clases padre e hijas, pero los hace no-visibles al resto del mundo.

CONSTRUCTORES

Un constructor es el método de una clase, que se ejecuta cuando usamos

"New clase" , es decir cuando creamos un objeto . El constructor puede tener 0 ó mas parámetros .

Todas las clases se crean con un constructor por defecto. Este constructor no tiene parámetros y básicamente no hace nada más que crear la clase.