Entendiendo Polimorfismo

El concepto base del polimorfismo no es que la capacidad que tienen las clases de responder a un método de otra clase de la cual lo heredan de una manera única y especifica de dicha clase que ja heredado, es decir, de una manera singular siguiendo las características de esta.

Esta forma ce implementación de métodos ya creados de maneras diferentes permiten que los códigos que se creen sean más comprensibles por las personas que los leen ya que, indirectamente, nos evita tener que identificar o rastrear de donde viene un método ya que, en estos casos siempre se puede conocer dicho método heredado y alterado por su nombre, el cual nunca cambia independientemente de las veces que este sea heredado.

Es un ejemplo claro, ciertas situaciones en las que tenemos más de una clase que posee el mismo método implementado en cada una de ellas, y dicho método tiene diferentes formas de representar su función a pesar de ser la misma, por ejemplo, tenemos un listado de piezas existentes en un almacén implementadas en una aplicación como clases y todas tienen un método llamado “obtenerInfoPieza”, y el código, dependiendo de lo que se solicite implementara dicho método con su aplicación específica para cada clase sin alterar los otros tipos de funciones que se ejecuten bajo su mismo nombre en otras clases.

Esto es útil ya que, en caso de agregar otros tipos de piezas que necesiten acceder a este método, incluir una clase para dichas piezas que lo agregue o implemente “obtenerInfoPieza”, no será un problema.

Puede darse la situación en que se cree una instancia como una lista de clases en las cuales se adiciona una clase para revisar las cuentas y, si creamos un método que cicle entre dichas clases, veremos, como hemos dicho anteriormente, que cada una de las clases tendrá su propia implementación del método “obtenerInfoPieza”.

En los polimorfismos también se nos permite usar polimorfismos mediante interfaces y herencias. Ahora estaremos viendo el uso de polimorfismos en herencias e interfaces.

Implementar polimorfismos usando herencias

Implementar polimorfismos usando herencia: para explicar la implementación de polimorfismos usando herencias estaremos usando de ejemplo un ejercicio, que tenga una clase abstracta base llamada animales. Nuestra clase tiene un constructor que nos pide el tipo de animal y un método abstracto que nos retorne que tipo de animal es.

Con nuestra clase creada ahora debemos crear dos subclases que hereden nuestra clase base, estas se llamaran mamífero y reptiles. Luego a estas clases les estaremos agregando nuestro método tipo de animal con un override para solucionar los problemas que esto nos pueda ocasionar y que no haya conflictos en el programa, ya que estos le dieron override a la clase tipo de animal del método base.

Implementar polimorfismos usando interfaces

La implementación de polimorfismos con el uso de interfaces es básicamente lo visto con las herencias solo que en este caso estaremos haciendo el ejemplo con interfaces y para este usaremos el caso de un ciudadano que este registrado en su ayuntamiento con sus datos.

 Creamos una interfaz de ciudadano a la cual le crearemos dos clases que se deriven de esta (estado laboral, estado civil). Ambas clases poseen los mismos métodos, en el main estaremos usando un for each en con la interfaz y las dos clases que acabamos de crear para que este código funcione.