P.O.O.

¿A que nos referimos cuando decimos Programación Orientada a Objetos?

La Programación Orientada a Objetos es una de las formas más populares de programar y viene teniendo gran acogida en el desarrollo de proyectos de software desde los últimos años. Esta acogida se debe a sus grandes capacidades y ventajas frente a las antiguas formas de programar. Para tener una idea más amplia de la P.O.O vamos a revisar 4 conceptos básicos.

- Objetos
- Clases
- Herencia
- Envío de Mensajes.

Objetos.

Entender que es un objeto es la clave para entender cualquier lenguaje orientado a objetos. Existen muchas definiciones que se le ha dado al Objeto. Empecemos entendiendo que es un objeto del mundo real, un Objeto del mundo real es cualquier cosa que vemos a nuestro alrededor, digamos que para leer un artículo en internet lo hacemos a través del monitor y una computadora, ambos son objetos, al igual que nuestro teléfono, una mesa, un árbol, un automóvil, etc.

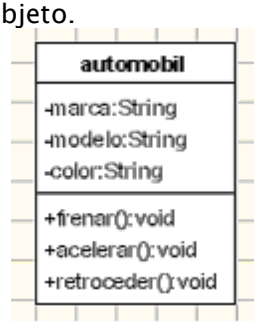
Analicemos un poco más a un objeto del mundo real, como la computadora, no necesitamos ser expertos en hardware para saber que una computadora está compuesta internamente por varios componentes: la tarjeta madre, el procesador, el disco duro, las memorias, y otras partes más. El trabajo en conjunto de todos estos componentes hace operar a una computadora.

Internamente, cada uno de estos componentes puede ser sumamente complicado y puede ser fabricado por diversas compañías con diversos métodos de diseño. Pero nosotros no necesitamos saber cómo trabajan cada uno de estos componentes, cada componente es una unidad autónoma, y todo lo que necesitamos saber de adentro es cómo interactúan entre sí los componentes, saber por ejemplo si el procesador y las memorias son compatibles con la tarjeta madre, o conocer donde se coloca la tarjeta de video. Cuando conocemos como interaccionan los componentes entre sí, podremos armar fácilmente una computadora.

¿Que tiene que ver esto con la programación? La programación orientada a objetos trabaja de esta manera. Todo el programa está construido en base a diferentes componentes (Objetos), cada uno tiene un rol específico en el programa y todos los componentes pueden comunicarse entre ellos de formas predefinidas.

Todo objeto del mundo real tiene 2 componentes: características y comportamiento. Por ejemplo, los automóviles tienen características (marca, modelo, color, velocidad máxima, etc.) y comportamiento (frenar, acelerar, retroceder, llenar combustible, cambiar llantas, etc.).

Los Objetos de Software, al igual que los objetos del mundo real, también tienen características y comportamientos. Un objeto de software mantiene sus características en una o más "variables", e implementa su comportamiento con "métodos". Un método es una función o subrutina asociada a unobjeto.

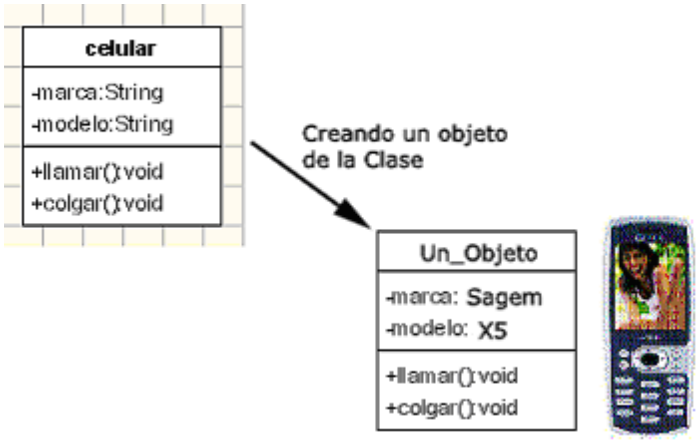


Para redondear estas ideas, imaginemos que tenemos estacionado en nuestra cochera un Ford

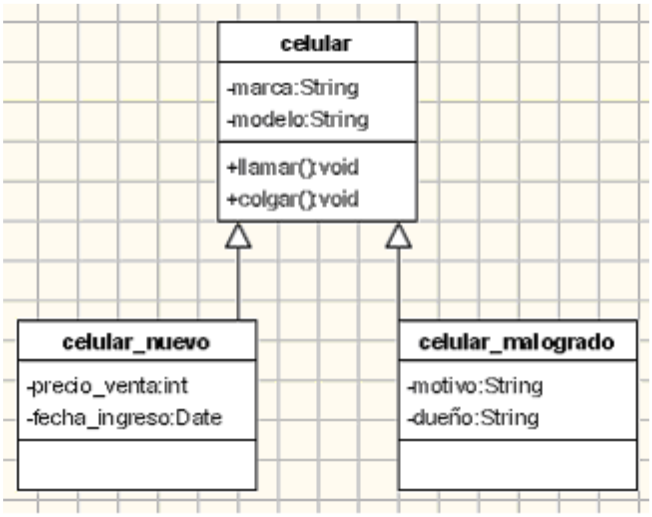
Focus color azul que corre hasta 260 km/h. Si pasamos ese objeto del mundo real al mundo del software, tendremos un objeto Automóvil con sus características predeterminadas:

Marca = Ford
Modelo = Focus
Color = Azul
Velocidad Máxima = 260 km/h

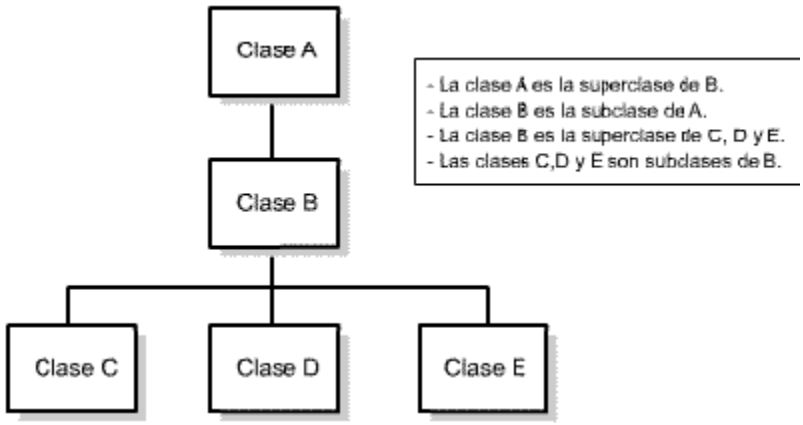
Cuando a las características del objeto le ponemos valores decimos que el objeto tiene estados. Las variables almacenan los estados de un objeto en un determinado momento.
Definición teórica: ***Un objeto es una unidad de código compuesto de variables y métodos relacionados.*** **Clases**
En el mundo real, normalmente tenemos muchos objetos del mismo tipo. Por ejemplo, nuestro teléfono celular es sólo uno de los miles que hay en el mundo. Si hablamos en términos de la programación orientada a objetos, podemos decir que nuestro objeto celular es una ***instancia de una clase*** conocida como "celular". Los celulares tienen características (marca, modelo, sistema operativo, pantalla, teclado, etc.) y comportamientos (hacer y recibir llamadas, enviar mensajes multimedia, transmisión de datos, etc.). Cuando se fabrican los celulares, los fabricantes aprovechan el hecho de que los celulares comparten esas características comunes y construyen modelos o plantillas comunes, para que a partir de esas se puedan crear muchos equipos celulares del mismo modelo. A ese modelo o plantilla le llamamos CLASE, y a los equipos que sacamos a partir de ella la llamamos OBJETOS.



Esto mismo se aplica a los objetos de software, se puede tener muchos objetos del mismo tipo y mismas características.
Definición teórica: La clase es un modelo o prototipo que define las variables y métodos comunes a todos los objetos de cierta clase. También se puede decir que una clase es una plantilla genérica para un conjunto de objetos de similares características. Por otro lado, una instancia de una clase es otra forma de llamar a un objeto. En realidad no existe diferencia entre un objeto y una instancia. Sólo que el objeto es un término más gen eral, pero los objetos y las instancias son ambas representación de una clase.
Definición Teórica: ***Una instancia es un objeto de una clase en particular.***
Herencia.
La herencia es uno de los conceptos más cruciales en la POO. La herencia básicamente consiste en que una clase puede heredar sus variables y métodos a varias subclases (la clase que hereda es llamada superclase o clase padre). Esto significa que una subclase, aparte de los atributos y métodos propios, tiene incorporados los atributos y métodos heredados de la superclase. De esta manera se crea una jerarquía de herencia.
Por ejemplo, imaginemos que estamos haciendo el análisis de un Sistema para una tienda que vende y repara equipos celulares.



En el gráfico vemos 2 Clases más que posiblemente necesitemos para crear nuestro Sistema. Esas 2 Clases nuevas se construirán a partir de la Clase Celular existente. De esa forma utilizamos el comportamiento de la SuperClase.
En general, podemos tener una gran jerarquía de Clases tal y como vemos en el siguiente gráfico:



Envío de Mensajes
Un objeto es inútil si está aislado. El medio empleado para que un objeto interactúe con otro son los mensajes. Hablando en términos un poco más técnicos, los mensajes son invocaciones a los métodos de los objetos.
Lenguajes de Programación Orientada a Objetos En 1985, E. Stroustrup extendió el lenguaje de programación C a C++, es decir C con conceptos de clases y objetos. En 1995 apareció el más reciente lenguaje OO, Java desarrollado por SUN, que hereda conceptos de C++.
El lenguaje de desarrollo más extendido para aplicaciones Web, el PHP, trae todas las características necesarias para desarrollar software orientado a objetos.
Además de otros lenguajes que fueron evolucionando, como el Pascal a Delphi.