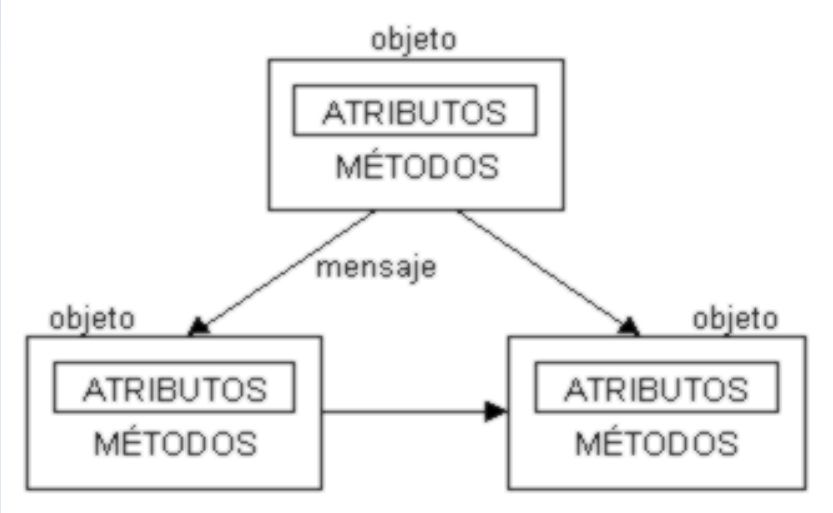
Paradigmas de Programación

PROGRAMACION ORIENTADA A OBJETOS



Moisés García Villanueva

5 DE OCTUBRE DE 2012

Programacion OO (POO)

- La POO viene de la evolución de la programación estructurada; básicamente la POO simplifica la programación con la nueva filosofía y nuevos conceptos que tiene.
- La POO se basa en dividir el programa en pequeñas unidades lógicas de código.
- A estas pequeñas unidades lógicas de código se les llama objetos.
- Los objetos son unidades independientes que se comunican entre ellos mediante mensajes.

Ventajas de POO

- Fomenta la reutilización y extensión del código.
- Permite crear sistemas más complejos.
- Relacionar el sistema al mundo real.
- Facilita la creación de programas visuales.
- Construcción de prototipos
- Agiliza el desarrollo de software
- Facilita el trabajo en equipo
- Facilita el mantenimiento del software

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.

- Definir un objeto real!!!!!!!!

- En la POO, 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.

Qué son los objetos

- Todo objeto del mundo real tiene dos componentes:
- CARACTERÍSTICAS
- COMPORTAMIENTO

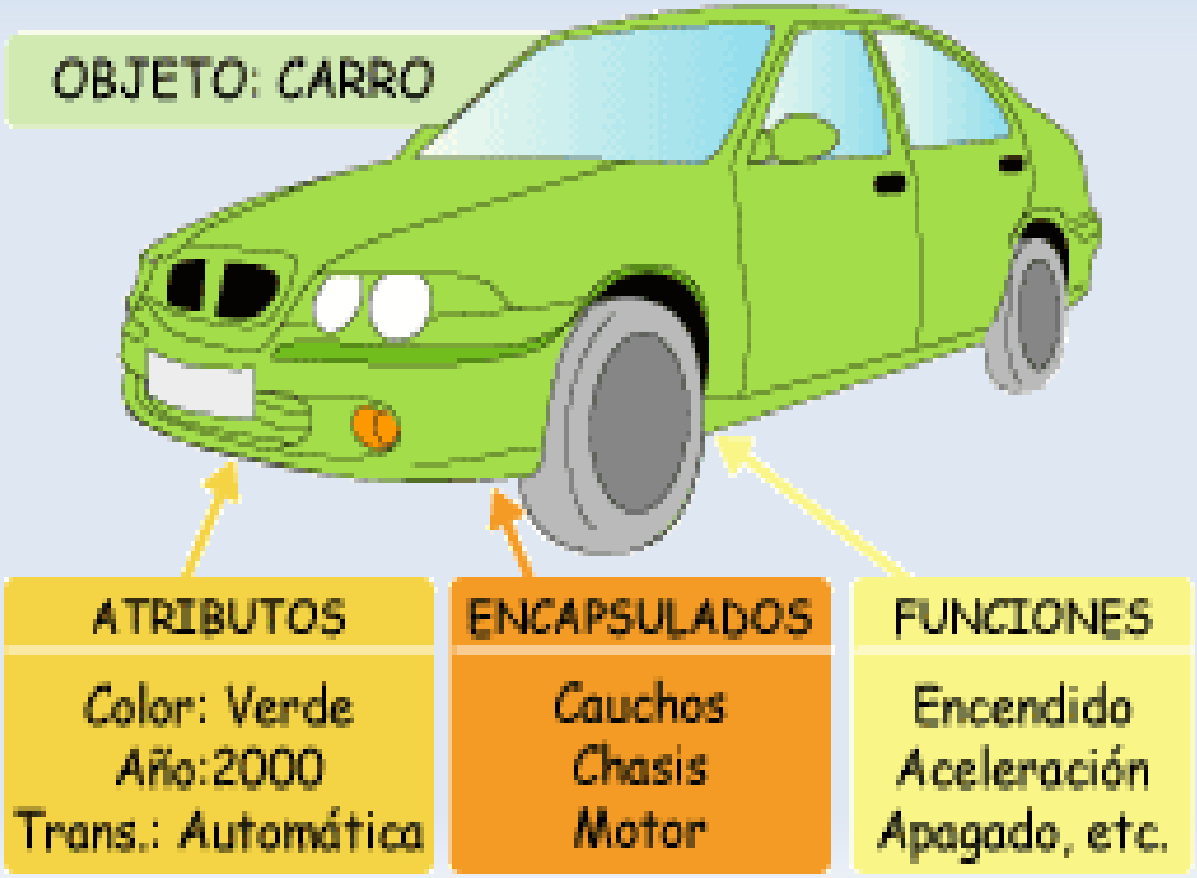
Mencionemos las características y comportamientos de objetos ¿como cuales se les ocurren?

Objetos de software

- 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"
- Implementa su comportamiento con "métodos".
- Un método es una función o subrutina asociada a un objeto.

Definición

- Un objeto es una unidad de código compuesto de variables y métodos relacionados.



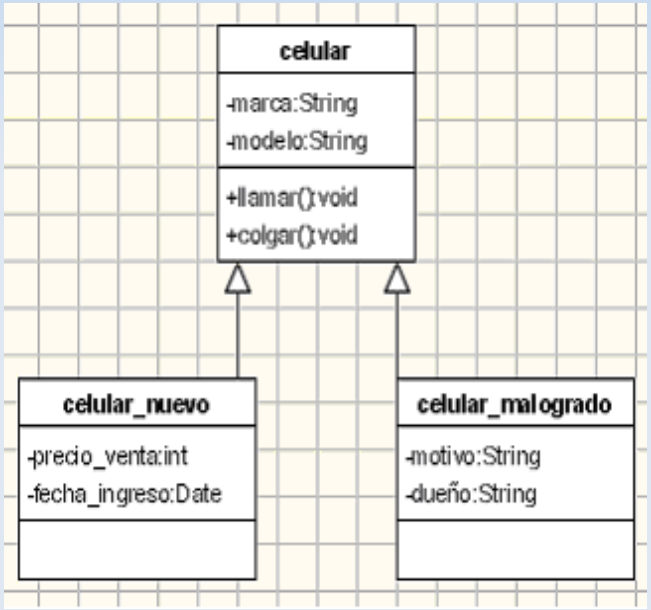
Clases

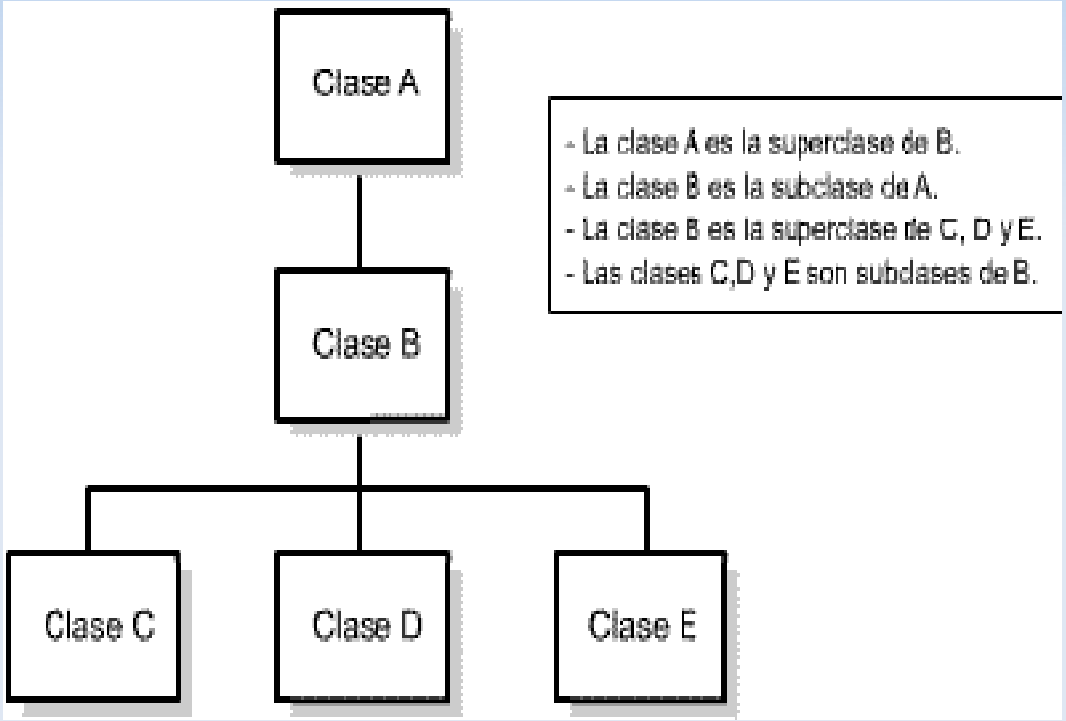
- Las clases son declaraciones de objetos, también se podrían definir como abstracciones de objetos.
- Esto quiere decir que la definición de un objeto es la clase.
- Cuando programamos un objeto y definimos sus características y funcionalidades en realidad lo que estamos haciendo es programar una clase.

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.

Herencia


celular
-marca:String
-modelo:String
+llamar():void
+colgar():void
celular_nuevo
-precio_venta:int
-fecha_ingreso:Date
celular_malogrado
-motivo:String
-dueño:String


Clase A
Clase B
Clase C
Clase D
Clase E
- La clase A es la superclase de B.
- La clase B es la subclase de A.
- La clase B es la superclase de C, D y E.
- Las clases C,D y E son subclases de B.

Características asociadas a la POO

- Abstracción

La abstracción consiste en captar las características esenciales de un objeto, así como su comportamiento.

- Encapsulamiento

El encapsulamiento consiste en unir en la Clase las características y comportamientos, esto es, las variables y métodos. Es tener todo esto es una sola entidad.

Encapsulamiento

- La utilidad del encapsulamiento va por la facilidad para manejar la complejidad, ya que tendremos a las Clases como cajas negras donde sólo se conoce el comportamiento pero no los detalles internos, y esto es conveniente porque nos interesará conocer qué hace la Clase pero no será necesario saber cómo lo hace.

Ocultamiento

- Es la capacidad de ocultar los detalles internos del comportamiento de una Clase y exponer sólo los detalles que sean necesarios para el resto del sistema.

- El ocultamiento permite 2 cosas:

  1) Restringir el uso de la Clase. Restringir porque habrá cierto comportamiento privado de la Clase que no podrá ser accedido por otras Clases.

  2) Controlar el uso de la clase. Controlar porque daremos ciertos mecanismos para modificar el estado de nuestra Clase y es en estos mecanismos dónde se validarán que algunas condiciones se cumplan.

En Java el ocultamiento se logra usando las palabras reservadas: public, private y protected delante de las variables y métodos.

Polimorfismo

- Es la capacidad de que diferentes objetos reaccionen de distinta forma a un mismo mensaje.
- Es la capacidad de referirse a objetos de clases distintas en una jerarquía utilizando el mismo elemento de programa (método) para realizar la misma operación, pero de manera diferente.

```
class Box {

  double width;

  double height;

  double depth;

    //El siguiente es el constructor específico

  Box(double w, double h, double d) {

    width = w; height = h; depth = d;   }

    //pero podría ser que no le llegarán parámetros   // por fallar la otra clase (método) que lo
    invoque

  Box () {

   width = height = depth = -1 //-1 indica volumen no existente   }

  //e incluso podemos pensar que se quiere construir un cubo, entonces, por qué introducir 3
    valores? ;)

  Box (double valor) {

    width = height = depth = valor;   }

    double volume() {

    return width * height * depth;

  } }
```

Análisis y diseño Orientado a Objetos

- Para el desarrollo de software orientado a objetos no basta usar un lenguaje orientado a objetos. También se necesitará realizar un análisis y diseño orientado a objetos.
- El modelamiento visual es la clave para realizar el análisis OO. Desde los inicios del desarrollo de software OO han existido diferentes metodologías para hacer esto del modelamiento, pero sin lugar a duda, el Lenguaje de Modelamiento Unificado (UML) puso fin a la guerra de metodologías.

Análisis y diseño Orientado a Objetos

Según los mismos diseñadores del lenguaje UML, éste tiene como fin modelar cualquier tipo de sistemas (no solamente de software) usando los conceptos de la orientación a objetos. Y además, este lenguaje debe ser entendible para los humanos y máquinas.

- Actualmente en la industria del desarrollo de software tenemos al UML como un estándar para el modelamiento de sistemas OO.

Ejercicio

- Crea una clase Hora con atributos para las horas, los minutos y los segundos de la hora Incluye, al menos, los siguientes métodos:
- Constructor predeterminado con el 00:00:00 como hora por defecto. En el constructor se podrán indicar horas, minutos y segundos.
- leer(): pedirá al usuario las horas, los minutos y los segundos.
- valida(): comprobará si la hora es correcta; si no lo es la ajustará. Será un método auxiliar (privado) que se llamará en el constructor parametrizado y en leer().
- a_segundos(): devolverá el número de segundos transcurridos desde la medianoche.
- de_segundos(int): hará que la hora sea la correspondiente a haber transcurrido desde la medianoche los segundos que se indiquen.
- segundos_desde(Hora): devolverá el número de segundos entre la hora y la proporcionada.
- siguiente(): pasará al segundo siguiente.
- anterior(): pasará al segundo anterior.
- copia(): devolverá un clon de la hora.
- Además (métodos especiales):
- print: mostrará la hora (07:03:21); igual_que(Hora): indica si la hora es la misma que la proporcionada; menor_que(Hora): indica si la hora es anterior a la proporcionada.
- mayor_que(Hora): indica si la hora es posterior a la proporcionada.