

UNIDAD 3. CARACTERÍSTICAS DE POO Y EXCEPCIONES

MATERIA: Programación orientada a objetos I



15 DE MAYO DE 2020

Guadalupe García Vázquez MATRICULA: ES1921008556 Desarrollo en Software

Docente: Claudia Erika González Gómez

Desarrollo de Software

Actividad 2: Herencia y Polimorfismo. Fase 2

Contenido

Introducción	3
Desarrollo	4
Conclusiones	10
Bibliografía	11

Desarrollo de Software

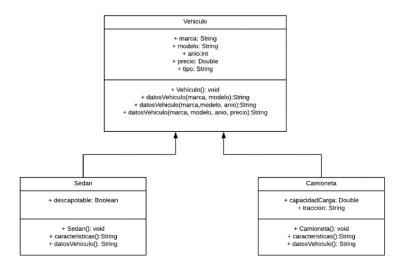
Introducción

El siguiente tema veremos los conceptos de Herencia, Superclase, subclase, y polimorfismo.

Aplicándolo a un ejercicio y programándolo en Netbeans, se detectará la superclase(clase padre), la subclase(clase hija), jerarquía de clases y polimorfismo.

Desarrollo

1. Analiza el siguiente diagrama de clases



2. Aplicando el concepto de herencia y polimorfismo, realice la codificación del diagrama de clases indicado en el paso 1.

Para lograr aplicar de manera adecuada los conceptos de Herencia y polimorfismo en Netbeands.

Primero creamos la Superclase "Vehiculos" con los atributos y métodos correspondientes.

Desarrollo de Software

```
DPO1_U3_A2_GUGV - Apache NetBeans IDE 11.3
 File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
🕆 🚰 🛂 🤚 🥱 🧖 | <default config> 🔻 🚳 🕆 🏋 🎉 👂 🕻 🐘 🗥 🖟 🔁 297.6/543.008
                    — ClasePrincipal, java × 📾 Vehiculo, java × 📾 Sedan, java × 📾 Camioneta, java ×
   Source Packages

Grandm.dpol_u3_a2_gu

Camioneta,java

GlasePrincipal,java

Welsiculo,java

Welsiculo,java

Test Parkwas

Test Parkwas

Test Parkwas
                            Source History | 🚱 👼 - 👼 - | 🦞 😓 😂 📮 📮 | 🌮 😓 | 월 월 | 🄞 📓 🕍 🚅
                                                                                                                                                                                                                 E Output
Test Packages
Dependencies
Dependencies
Dava Dependencies
Project Files
UMLDiagrams
    Test Packages
                                           protected String marca;
                                          protected String modelo;
protected int anio;
                                     protected Double precio;
                                              //Metodos
                              14
15
16
17 =
18
19
20
21
                                           //Metodo constructor-mismo nombre clase, no devuelve valor, declararse como publico
                                         public Vehiculo (String marca, String modelo, int anio, Double precio, String tipo) (
                                           this.marca=marca;
this.modelo= modelo;
this.anio= anio;
                                           this.precio= precio;
                              22
23
24
                                           this.tipo= tipo;
                              25
26
27
28
                                          //Getter y Setter
                                          public String getMarca() {
                              29
30
                                                return marca
                                          public void setMarca(String marca) (
```

Así mismo se utilizaron los métodos Get y Set. Los métodos get y set, son simples métodos que usamos en las clases para mostrar (get) o modificar (set) el valor de un atributo, esto con la finalidad de que los valores de las clases estuvieran protegidos.

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
                                                                                                                                                                  Q Search (Ctrl+I)
 = SclasePrincipal, java × Schiculo, java × Schan, java × Schan, java × Camioneta, java ×
                        Source History | 🚱 💀 - 👼 - | 🔾 😎 🖶 📮 | 🔗 😓 | 😉 💇 | 🥚 🗎 | 😃 🚅
                                                                                                                                                                                         H 15
   DPO1_US_AZ_GUGV
Source Packages
unadm.dpo1_u3_a2_gu
Ø Camioneta,java
Ø Sedan,java
Ø Vehiculo,java
Penendencies
                          57
58
59
60 =
                                          this.precio = precio;
                                    public String getTipo() {
                                    return tipo;
                           61
62
63
Test Packages
Dependencies
Dependencies
Java Dependencies
No. Project Files
UMLDiagrams
                           64 📮
                                     public void setTipo(String tipo) (
                                          this.tipo = tipo;
                                     //Hetodos retornos valores
                           public String datosVehiculo() (
                          76
77
78
79
80
81
82
                                                 "\nMarca: "+marca+"\nModelo: "+modelo+"\nAño: "+anio+"\nPrecio:$ "+precio;
                                     }//datosvehiculo
                                      }//Vehiculo
    < >
```

Desarrollo de Software

Subclase Sedan,

```
DPO1_U3_A2_GUGV - Apache NetBeans IDE 11.3
 File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
🕆 🚰 🛂 🤚 🥱 🧖 | <default config> 🔻 🚳 🕆 🏋 🎉 🖟 - 🎊 🐘 - 🕦 - | 435.81543.006 🗘 😘
                   ₽ Projects ×
                          Source History | 😭 🔯 - 📓 - | 🔾 😎 🚭 📮 | 🚱 😓 | 😭 💇 | 🍏 | 📓 🕌 📑
   Source Packages
                                  package unadm.dpo1 u3 a2 gugv;
    🗐 🕮 unadm.dpo1 u3 a2 gu
        Camioneta,java
ClasePrincipal,java
Sedan,java
Vehiculo,java
                                   public class Sedan extends Vehiculo(
                                     protected Boolean descapotable;
                             protected sociean descapotable;

//inicioconstructor

public Sedam(String marca, String modelo, int anio, Double precio, String tipo, Boolean descapotable)(
super(marca, modelo, anio, precio, tipo);
//instancia de los atributos de la clase hija
Test Packages
Dependencies
Java Dependencies
WMLDiagrams
                                        this.descapotable=descapotable;
                                    } //finalconstructor
                            15
14
15
16 =
17
18
19 =
                                     //getter y setter
public Boolean getDescapotable() (
                                           return descapotable;
                                        public void setDescapotable(Boolean descapotable) (
                                             this.descapotable = descapotable;
                            )//Fin getter
                                         //sobreescribimos el metodo de la clase padre, ya que aparte de tener los at<mark>r</mark>ibutos de la clase padre tiene un atributo más
                                     @Override
                                       public String datosVehiculo() (
                                         return "\n\n. ..."
"\nDatos del Sedan"+
                            28
                                                      "\nMarca: "+marca+"\nModelo: "+modelo+"\nAño: "+anio+"\nPrecio:$"+precio+
                                              "\nCaracterísticas del Sedan: "+
"\nDescapotable:"+descapotable;
                            29
30
                            31
```

Subclase Camioneta

```
DPO1_U3_A2_GUGV - Apache NetBeans IDE 11.3
  File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
                                                                                                                                                                                                                                                                                                                                                                                                        Q Search (Ctrl+I)
  🚰 🚰 🛂 🦃 🧖 (default config> 🔻 🚳 🕆 🧸 🖟 🖟 🖟 🖟 🖟 🖒 🖟
                                             _ GlasePrincipal,java × 📾 Vehiculo,java × 📾 Sedan,java × 📾 Camioneta.java ×
Projects X
B DPD1_U3_R2_GUGV
Source
Source Packages
G Camioneta_java
G CasePrincpal_java
G Sedan_java
G Sedan
                                                           Source History | 🚱 👨 - 👼 - | 🔾 😎 👺 🖶 📮 | 🔗 😓 | 🖄 🖄 | 🚳 🗎 | 💇 🚅
                                                                              package unadm.dpo1_u3_a2_gugv;
                                                                               public class Camioneta extends Vehiculo(
                                                                                 protected Double capacidadCarga;
protected String traccion;
         Test Packages
Test Packages
Dependencies
Java Dependencies
Java Dependencies
UMLDiagrams
                                                                  8 public Camioneta(String marca,String modelo,int anio,Double precio,String tipo,Double capacidadCarga,String traccion)(
                                                                                            this.capacidadCarga=capacidadCarga;
                                                                 11
                                                                                           this.traccion=traccion;
                                                                                  } //finalconstructor
                                                                 public Double getCapacidadCarga() {
    return capacidadCarga;
                                                                                            public void setCapacidadCarga(Double capacidadCarga) {
                                                                                                    this.capacidadCarga = capacidadCarga;
                                                                                           public String getTraccion() (
                                                                 23
                                                                                                      return traccion;
                                                                24
25
26
27
                                                                                            public void setTraccion(String traccion) {
                                                                                                 this.traccion = traccion;
                                                                                      //sobreescribimos el metodo de la clase padre, ya que aparte de tener los atributos de la clase padre tiene un atributo más
                                                                                      @Override
                                                                                   public String datosVehiculo() {
                                                                                             return "\n\n....
                                                                                                                                                                                                                                                                                                                                                                                                                          2
                                                                                                                                                                                                                                                                                                                                                                                                                                               26:34
```

En este puntos nos encontramos con un problema.

Desarrollo de Software

Cómo podemos observar en la siguiente imagen Netbeans nos marcaba un error, ya que estamos utilizando el método de la clase Padre, se escribe @Override para poder sobreescribir el método.

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

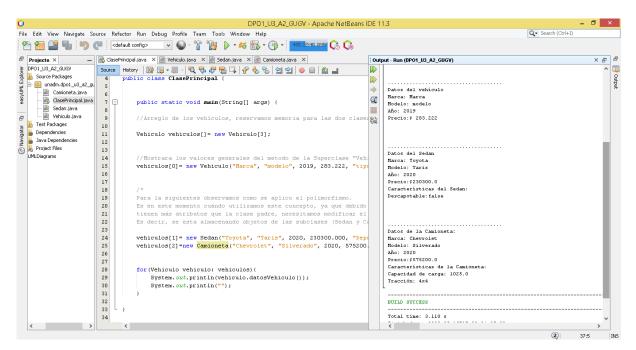
| Comparison |
```

Desarrollo de Software

Y por último en la Clase principal, instanciamos las subclases.

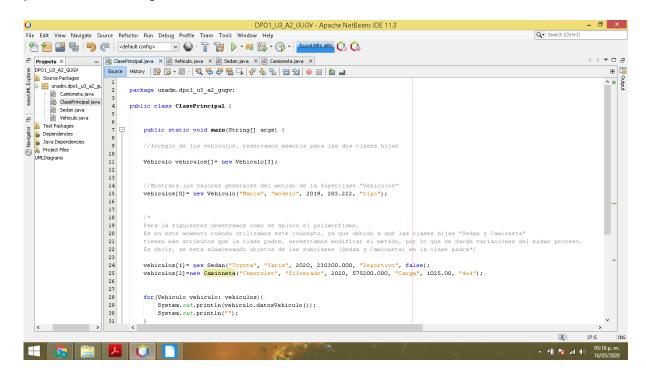
```
DPO1_U3_A2_GUGV - Apache NetBeans IDE 11.3
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
🚰 🚰 🛂 🦃 🌎 🍘 🔀 <default config> 🔻 🚳 🕆 🥳 🖟 🕬 🔻 377.0/543.0MB 🗘 😘
                      _ Sedan.java × Sedan.java × Camioneta.java ×
                          Source Packages
    🗐 📖 unadm.dpo1 u3 a2 gu
       Camioneta.java
ClasePrincipal.java
Sedan.java
Vehiculo.java
                             6
7 📮
                                         public static void main(String[] args) (
                            8
9
10
                                        //Arreglo de los vehículos, reservamos memoria para las dos clases hijas
   Test Packages
Dependencies
Java Dependencies
Project Files
UMLDiagrams
                            11
                                        Vehiculo vehiculos[]= new Vehiculo[3]:
                            12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
                                         vehiculos[0] = new Vehiculo("Marca", "modelo", 2019, 283.222, "tipo");
                                         Para la siguientes observamos como se aplico el polimorfismo
                                        Es en este momento cuándo utilizamos este concepto, ya que debido a qué las clases hijas "Sedan y Camioneta" tienen más atributos que la clase padre, necesitamos modificar el metodo, por lo que se darán variaciones del mismo proce
                                         Es decir, se esta almacenando objetos de las subclases (Sedan v Camioneta) en la clase padre*/
                                        vehiculos[1]= new Sedan("Toyota", "Yaris", 2020, 230300.000, "Deportivo", false);
vehiculos[2]=new Camioneta("Chevrolet", "Silverado", 2020, 575200.000, "Carga", 1025.00, "4x4");
                                             System.out.println(vehiculo.datosVehiculo());
                                             System.out.println("");
                            34
```

Resultando lo siguiente. La primera representa el método del padre, sin ninguna modificación. La segunda hacemos el llamado a las Subclases de Sedan y Camioneta, con la modificación del método de la superclase, utilizando el polimorfismo.



Desarrollo de Software

3. Describa dentro del código de su proyecto (como comentario) donde aplica herencia y polimorfismo, entregue sus clases en Java.



Desarrollo de Software

Conclusiones

Como vimos de la actividad anterior la herencia y el polimorfismo son dos conceptos que van unidos. Ambos nos facilitan la programación, siendo dos temas fundamentales dentro de la programación orientada a objetos (POO).

Ahora lo pusimos en práctica y me ayudo a comprender como se utilizan estos conceptos.

En el ejercicio primero detecte cuál era la superclase, en este caso es Vehículos y las subclases Sedan y Camioneta.

La superclase vehículo heredará los métodos y atributos a sus hijas (Sedan y Camioneta) Pero estás tienen atributos independientes de la clase padre, más específicos de estás, por lo que la aplicación del método de la clase padre deber modificarse. Es cuando entra el polimorfismo.

Desarrollo de Software

Bibliografía

- Programación ATS. (Consultado 14/05/20). Curso de programación de Java. Obtenido de: Obtenido de: https://www.youtube.com/watch?v=2ZXiuh0rg3M&list=PLWtYZ2ejMVJkjOuTCzlk61j7XKfpIR74K
- Soria, P. R. (Consultado 14/05/20). *Herencia*. Obtenido de Programación orientada a objetos-Java: Obtenido de: https://catedu.gitbooks.io/programacion-orientada-a-objetosjava/content/herencia.html
- Universidad Abierta y a Distancia de México. (Consultado 14/05/20). *Unidad 3. Características de POO y excepciones*. Obtenido de Obtenido de: file: /DPO1_U3_Contenido.pdf
- Universidad Nacional Autónoma de México. (Consultado 14/05/20). *Programando con Java para aplicaciones Android.* Obtenido de Obtenido de: https://www.coursera.org/lecture/programandoconjava/herencia-y-polimorfismo-D8J6y
- Universidad Nacional Autónoma de México. (Consultado 14/05/2020). Herencia y Polimorfismo. Obtenido de Programando con Java para aplicaciones Android: Obtenido de: https://www.coursera.org/lecture/programandoconjava/herencia-y-polimorfismo-D8J6y

https://www.youtube.com/watch?v=LZPafq9Ve_Q&list=PLWtYZ2ejMVJkjOuTCzlk61j7XKfplR74K&index=75

https://www.youtube.com/watch?v=vQZ4RY 4I-

A&list=PLWtYZ2ejMVJkjOuTCzlk61j7XKfpIR74K&index=76

https://www.youtube.com/watch?v=RXcUTWczbag&list=PLWtYZ2ejMVJkjOuTCzlk61j7XKfpIR74K&index=78