Interfaces

Autor(es):

Gustavo Eduardo Gil Prado

Docente TC

Guía de Curso – 30/05/2022

Resumen:

El objetivo de esta presentación es presentar el conceptos de Interfaces como elemento fundamental dentro de la Programación Orientada a Objetos

Descripción:

Esta presentación tienen la intensión de servir de apoyo al desarrollo del curso de Programación Orientada a Objetos

Palabras clave:

Algoritmo, Clase, Instancia, Aplicación, Atributo, Método

FACULTAD DE INGENIERÍA

Ingenieria Informática

Institución Universitaria Colegio Mayor del Cauca

Referencie este documento así: Gil, G.E. (2023). Interfaces [Guia de Curso]. Institución Universitaria Colegio Mayor del Cauca.

Programación Orientada a Objetos

Interfaces

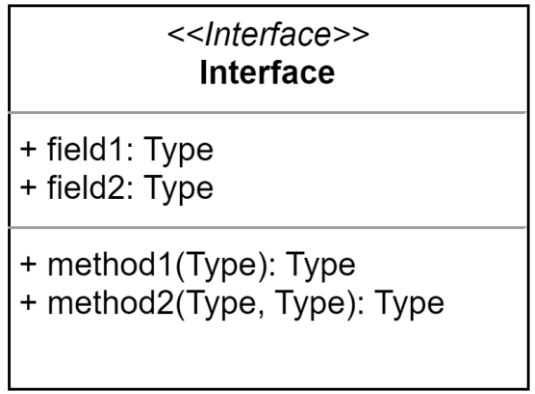
Docente: Gustavo Eduardo Gil Prado

Interfaces.

En POO, una interface es la descripción de uno o más métodos que deben ser implementados por una clase.

Es una clase que solo puede tener:

- Métodos abstractos y
- Atributos constantes



Declaración

JAVA:

Las palabras reservadas utilizadas en para la el trabajo con interfaces son:

- Interface: que se agrega en la declaración
- Implements: lo que permite implementar la interface por otra clase, la que puede ser abstracta o concreta.

C#

En este lenguaje, se emplean los dos puntos (:) como indicación de la implementación de la interface

Declaración Java-C#

Estructura:

```
1. Modificador_acceso interface NombreInterfaz
2. {
3.     //definición del contenido de la interfaz
4. }
```

Declaración Java – Uso en una clase.

Para utilizarlo en una clase:

1. modificador_acceso class NombreClase implements NombreInterfaz1 [, NombreInterfaz2]
2. {
3. .....//Contenido de la clase el que debe incluir la implementación de los métodos de todas las interfaces.
4. }

Interfaces

No evitan la duplicidad de código, al no aportar una implementación, así como no restringir sus posibles implementaciones.

No favorece la reutilización e código, por ovias razones

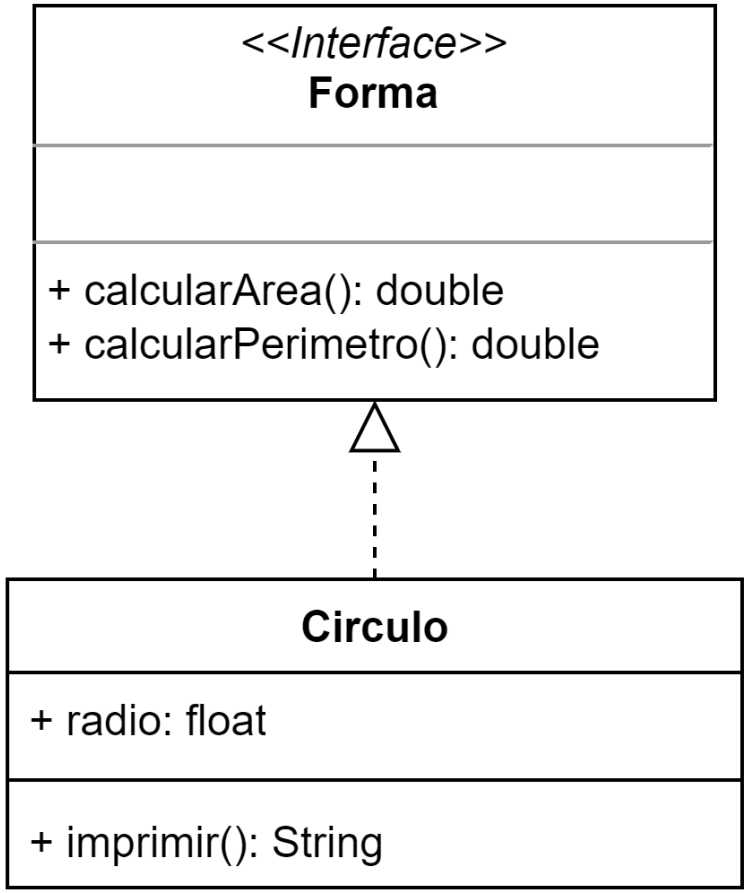
Interfaces – Ventajas

El uso de estas nos aportan a las soluciones software:

1. Organización, agregando convenciones a la codificación. Lo que favorece su mantención y extensión.
2. Favorece la existencia de miembros, especialmente de métodos polimórficos.
3. Permite tener valores constantes a disposición de las clases que las implementen

Interfaces – representación UML

Para la representación se utiliza la etiqueta <<interface>>, y para implementar una interface se tiene



Programación Orientada a Objetos

Clases internas

Docente: Gustavo Eduardo Gil Prado

Clases internas

En java es posible, definir una clase dentro de otra (anidada), lo que nos permite:

- Agrupar funcionalidades, separando responsabilidades.
- Aumentar el grado de encapsulación de la solución
- Hacer un código mas mantenible y legible.

Este tipo de clase son las que son anidadas y no estática.

Ejemplo

```
class ClaseExterna {
  int x = 10;


  private class ClaseInterna {
    int y = x+10;
  }


public static void main(String[] args) {
    ClaseExterna externa = new ClaseExterna();
    ClaseExterna.ClaseInterna interna =
                                    externa. new ClaseInterna ();
    System.out.println(interna.y);
  }
}
```

En el ejemplo podemos ver que:

- La clase interna tiene acceso a los atributos de la que la contine.
- Como miembros de la clase, estas pueden ser declaradas con niveles de acceso. (public, protected o private)
- Estas están asociadas a la instancia de la clase anfitriona.

Consideraciones de clases anidades

En algunos casos si solo necesitamos requerimos implementar una interfaz definiendo una única instancia de clase se puede utilizar una clase interna anónima.

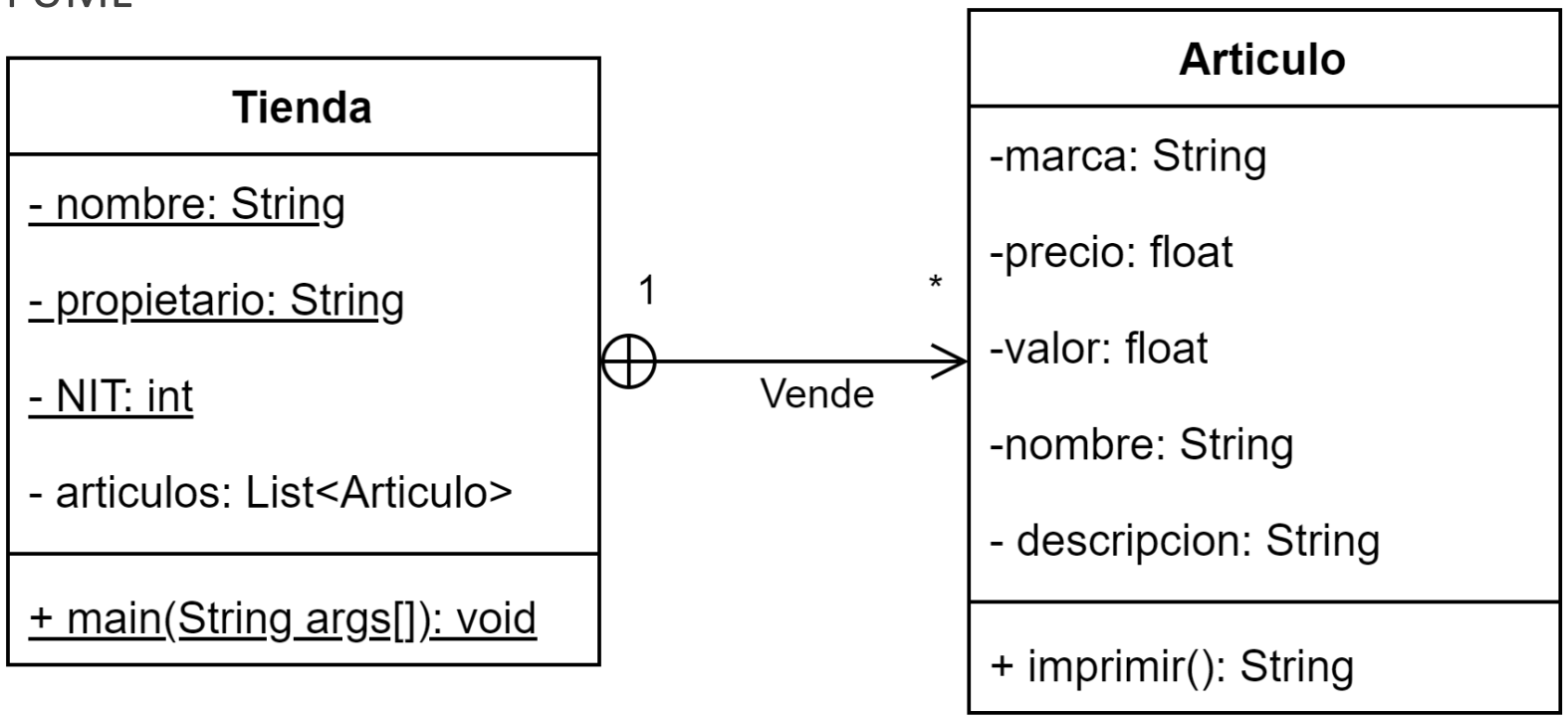
```
public class NewMain {
public interface interface_saludar{
      String TEXTO ="Hola ";
      public void hablar();
   }

   public static void main(String[] args) {

       interface_saludar p= new interface_saludar(){
          @Override
          public void hablar() {
             System.out.println(TEXTO+" soy pepe!!!");
          }
       };
       p.hablar();
   }
}
```

Clases internas

Representación UML



Ejercicio propuesto

- Defina mediante una clase interna si un atributo es positivo, negativo o cero, mediante una clase interna.

Utilizando clases internas, realice el calculo de las operaciones básicas (suma, resta, multiplicación y división) entre 2 atributos definidos en la clase externa.

Implemente una clase que contenga como atributo un arreglo de números, e indique mediante una clase interna, cual es numero mayor, el menor y el promedio de los valores almacenados en el arreglo.

Trabajo de casa – Siguiente lab

Complemente lo revisado en clase mediante una investigación de las principales características de las interfaces y las clases internas en Java.

Mediante un ejemplo construido con sus compañeros de equipo diseñe y realice la implementación de un ejercicio práctico, el que deben explicar a sus compañeros