UNIVERSIDAD TECNOLÓGICA DE HONDURAS

Asignatura:

Programación Orientada a Objetos

Catedrático:

ARNOL RAFAEL GUTIERREZ ALFARO

Estudiante:

Roberth Joel Lagos Sánchez

Cuenta:

202320110268

Modalidad:

Virtual

Sección 2

Segundo Parcial

Fecha: **01 de Noviembre 2024**

Documentación del Código: Diagrama de Formas

Introducción

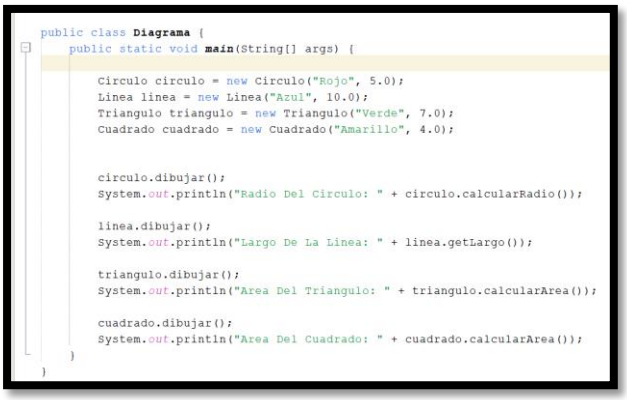
Explicación detallada del funcionamiento y la estructura del código en Java Netbeans, implementa un sistema para dibujar algunas formas geométricas. El programa crea instancias de las formas como ser: Circulo, Línea, Triangulo y Cuadrado, permitiendo visualizar sus características y calcular ciertos valores como el área y el radio.

Estructura del Código

El código está organizado en varias clases, cada una con un propósito específico. La clase principal se llama Diagrama, que se encarga de crear instancias de las formas y ejecutar métodos para dibujarlas y mostrar información.

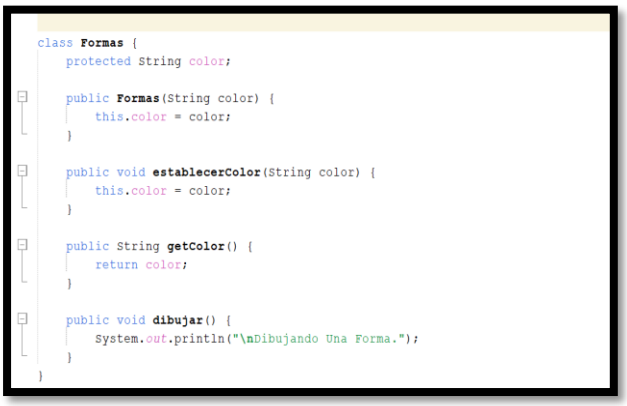
Clase Principal: Diagrama

En la clase Diagrama, se crean instancias de formas con distintos colores y se llaman sus métodos dibujar() para mostrar su representación en la consola. Además, se calculan valores como radio, largo y área mediante métodos específicos.



```
public class Diagrama {
    public static void main(String[] args) {

        Circulo circulo = new Circulo("Rojo", 5.0);
        Linea linea = new Linea("Azul", 10.0);
        Triangulo triangulo = new Triangulo("Verde", 7.0);
        Cuadrado cuadrado = new Cuadrado("Amarillo", 4.0);


        circulo.dibujar();
        System.out.println("Radio Del Circulo: " + circulo.calcularRadio());

        linea.dibujar();
        System.out.println("Largo De La Linea: " + linea.getLargo());

        triangulo.dibujar();
        System.out.println("Area Del Triangulo: " + triangulo.calcularArea());

        cuadrado.dibujar();
        System.out.println("Area Del Cuadrado: " + cuadrado.calcularArea());
    }
}
```

La superclase Formas: Actúa como una clase base para todas las formas geométricas. Contiene un atributo protegido color, que se establece a través del constructor.



```
class Formas {
    protected String color;

    public Formas(String color) {
        this.color = color;
    }

    public void establecerColor(String color) {
        this.color = color;
    }

    public String getColor() {
        return color;
    }

    public void dibujar() {
        System.out.println("\nDibujando Una Forma.");
    }
}
```

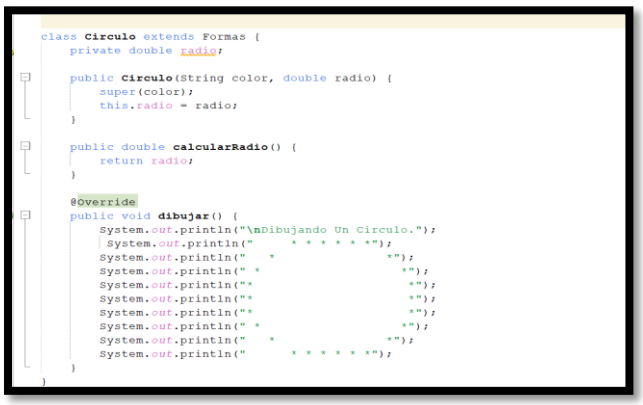
Subclases: Circulo, Línea, Triangulo y Cuadrado

Cada forma específica extiende la clase Formas y añade su propia funcionalidad.

1. Circulo.

Atributo: radio. **Métodos:** calcularRadio() que retorna el valor del radio.

Sobrescribe dibujar() para construir un círculo en la consola.



```
class Circulo extends Formas {
    private double radio;

    public Circulo(String color, double radio) {
        super(color);
        this.radio = radio;
    }

    public double calcularRadio() {
        return radio;
    }

    @Override
    public void dibujar() {
        System.out.println("\nDibujando Un Circulo.");
        System.out.println("      * * * * * *");
        System.out.println("    *             *");
        System.out.println("  *                 *");
        System.out.println("*                     *");
        System.out.println("*                     *");
        System.out.println("*                     *");
        System.out.println("  *                 *");
        System.out.println("    *             *");
        System.out.println("      * * * * * *");
    }
}
```

2. Línea.

Atributo: largo. **Método:** getLargo() que retorna el largo de la línea.

Sobrescribe dibujar() para mostrar una línea horizontal.

```
class Linea extends Formas {
    private double largo;

    public Linea(String color, double largo) {
        super(color);
        this.largo = largo;
    }

    public double getLargo() {
        return largo;
    }

    @Override
    public void dibujar() {
        System.out.println("\nDibujando Una Linea.");
        System.out.println("_____________________");
    }
}
```

3. Triangulo.

Atributo: Base y Altura. **Método:** calcularArea() que calcula el área del triángulo

Sobrescribe dibujar() para construir un triángulo.

```java
class Triangulo extends Formas {
    private double Base;
    private double Altura;

    public Triangulo(String color, double base, double altura) {
        super(color);
        this.Base = Base;
        this.Altura = Altura;
    }

    public double calcularArea() {
        return 0.5 * Base * Altura;
    }

    @Override
    public void dibujar() {
        System.out.println("\nDibujando Un Triangulo.");
         System.out.println("         *");
        System.out.println("      *    *");
        System.out.println("    *        *");
        System.out.println("  *            *");
        System.out.println("* * * * * * * * *");

    }
}
```

4. Cuadrado:

Atributo: area. **Método:** calcularArea() que retorna el área del cuadrado.

Sobrescribe dibujar() para construir un cuadrado.

```java
class Cuadrado extends Formas {
    private double area;

    public Cuadrado(String color, double area) {
        super(color);
        this.area = area;
    }

    public double calcularArea() {
        return area;
    }

    @Override
    public void dibujar() {
        System.out.println("\nDibujando Un Cuadrado.");
         System.out.println("* * * * * *");
        System.out.println("*          *");
        System.out.println("*          *");
        System.out.println("*          *");
        System.out.println("* * * * * *");
    }
}
```

Funcionamiento del Código

El código comienza en la clase Diagrama, donde se crean instancias de las subclases Circulo, Línea, Triangulo y Cuadrado. Estas instancias llaman al método dibujar() y a otros métodos de cálculo para mostrar información en la consola. Cada forma geométrica tiene su propio método dibujar(), que imprime una representación visual simple. También, cada forma puede calcular su área y características específicas, ejemplificando la herencia y el polimorfismo en programación orientada a objetos.

Conclusión

Este código demuestra la implementación de un sistema básico de formas geométricas utilizando principios de programación orientada a objetos. La estructura clara y la separación de responsabilidades entre la superclase y las subclases permiten un fácil mantenimiento y expansión del código en el futuro.