Rev 1.1 20160118

Programación II

Tema 6. Polimorfismo

Contenido

1 Introducción

En este tema se trata el polimorfismo. Se trata de que un programa pueda manejar objetos sin estar seguro de a qué clase pertenecen exactamente. Para que esto pueda funcionar, tiene que estar presente el enlace dinámico o tardío.

2 Enlace tardío

El enlace tardío consiste en que el enlace entre la llamada a un método y el método en sí se produzca en tiempo de ejecución.

Ejemplo

```
import java.util.Random;

public abstract class Figura {
	public abstract double calculaArea();
}

public class Circulo extends Figura {
	private double radio;

	public Circulo(double r)
	{
		radio = r;
	}

	public double getRadio()
	{
		return radio;
	}

	@Override
	public double calculaArea()
	{
		return radio * radio * Math.PI;
	}

	@Override
	public String toString()
	{
		return String.format( "Circulo de radio %.2f", getRadio() );
	}
}
```

```
public class Rectangulo extends Figura {
	private double lado1;
	private double lado2;

	public Rectangulo(double l1, double l2)
	{
		lado1 = l1;
		lado2 = l2;
	}

	public double getLado1()
	{
		return lado1;
	}

	public double getLado2()
	{
		return lado2;
	}

	@Override
	public double calculaArea()
	{
		return getLado1() * getLado2();
	}

	@Override
	public String toString()
	{
		return String.format( "Rectangulo de lados %.2f y %.2f",
				getLado1(), getLado2() );
	}
}

public class Ppal {
	public static void main (String[] args)
	{
		Figura f;
		Random rnd = new Random( System.currentTimeMillis() );

		if ( rnd.nextInt( 10 ) > 5 ) {
			f = new Rectangulo( 5, 6 );
		} else {
			f = new Circulo( 3 );
		}

		System.out.println( "Superficie: " + f.calculaArea() );
	}
}
```

La clave del ejemplo está en el método *main()*. En primer lugar, se crea un generador de números aleatorios, y según un valor generado al azar entre 0 y 9, se crea un objeto **Círculo** o **Rectángulo**. Finalmente, se calcula el área de la figura creada.

Así, es imposible que en el momento de compilar *f.calculaArea()*, Java sepa si debe enlazarla con *Rectangulo.calculaArea()* o con *Circulo.calculaArea()*. Este enlace solo puede producirse en tiempo de ejecución, y por eso se conoce como dinámico o tardío.

En otros lenguajes de programación, es necesario indicar qué llamadas son susceptibles de utilizar enlace dinámico. En todo caso, en Java es posible solamente lo contrario: indicar qué métodos nunca van a utilizar enlace dinámico. En el ejemplo anterior, *getRadio()*, *getLado1()* o *getLado2()* son candidatos a no utilizar enlace dinámico, por lo que se les puede aplicar el modificador *final*.

Ejemplo

```
public abstract class Figura {
	public abstract double calculaArea();
}

public class Circulo extends Figura {
	private double radio;

	public Circulo(double r)
	{
		radio = r;
	}

	public final double getRadio()
	{
		return radio;
	}

	@Override
	public double calculaArea()
	{
		return radio * radio * Math.PI;
	}

	@Override
	public String toString()
	{
		return String.format( "Circulo de radio %.2f", getRadio() );
	}
}
```

```
public class Rectangulo extends Figura {
	private double lado1;
	private double lado2;

	public Rectangulo(double l1, double l2)
	{
		lado1 = l1;
		lado2 = l2;
	}

	public final double getLado1()
	{
		return lado1;
	}

	public final double getLado2()
	{
		return lado2;
	}

	@Override
	public double calculaArea()
	{
		return getLado1() * getLado2();
	}

	@Override
	public String toString()
	{
		return String.format( "Rectangulo de lados %.2f y %.2f",
					getLado1(), getLado2() );
	}
}
```

3 Polimorfismo

El polimorfismo consiste en poder tratar con objetos sin conocer exactamente a qué clases pertenecen.

Ejemplo

```
public class Ppal {
	// …
	public static void visualizaArea(Figura[] figuras)
	{
		for(int i = 0; i < figuras.length; ++i) {
			System.out.println( figuras[ i ].calculaArea() );
		}
	}
}
```