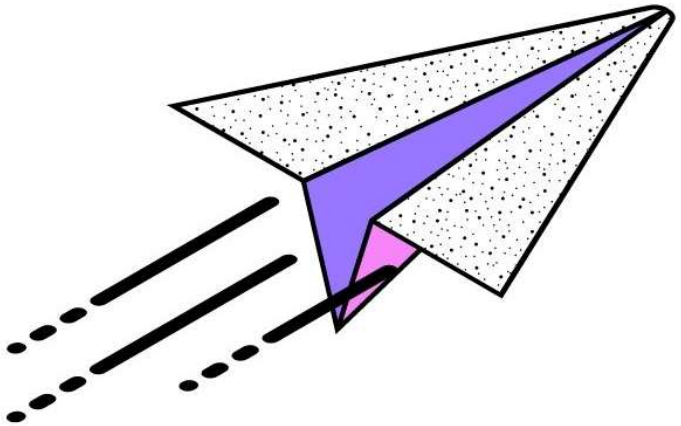
Java 7 SE Fundamentals

Seccion 10

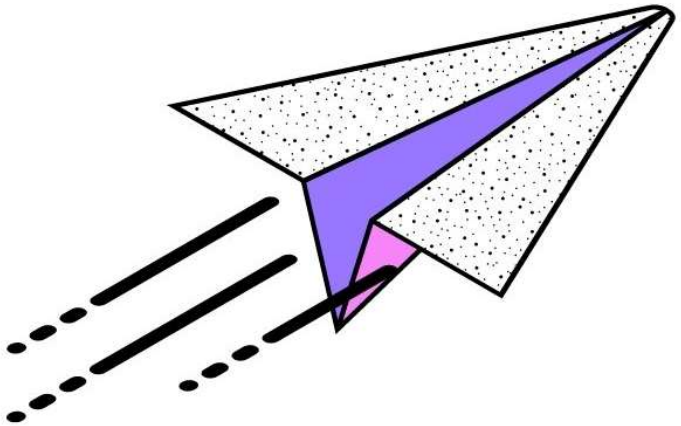
Conceptos avanzados del paradigma orientado a objetos



- Agregando abstracción al análisis y diseño
- Crear e implementar una Java Interface
- Uso de la herencia
- Comprender el propósito de las Java Interfaces.
- Uso del polimorfismo, sobreescritura
- Uso de superclases y subclases

Interfaces

conceptos clave



Que es

Una interfaz es similar a una clase, esta contiene solo constantes y métodos abstractos.

Proposito

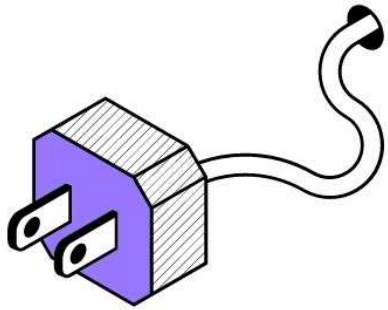
Su intención es especificar un comportamiento común para objetos de clases relacionadas o clases no relacionadas.

Como se implementa en Java

Para distinguir una interfaz de una clase, Java usa la siguiente sintaxis para definir una interfaz:

```
modifier interface InterfaceName {
        /** Constant declarations */
       /** Abstract method signatures */
}
```

Herencia



Conceptos clave

- La programación orientada a objetos permite definir nuevas clases a partir de clases existentes. Esto se llama herencia.
- La herencia le permite definir una clase general (por ejemplo, una superclase) y luego extenderla a clases más especializadas (por ejemplo, subclases).
- Una clase para modelar objetos del mismo tipo. Diferentes clases pueden tener algunas propiedades y comportamientos comunes, que pueden generalizarse en una clase que puede ser compartida por otras clases

Como se lleva a cabo la herencia en Java

- Naturalmente se necesitan dos clases, una una super clase y una subclase. La subclase heredara los métodos y atributos de la superclase.

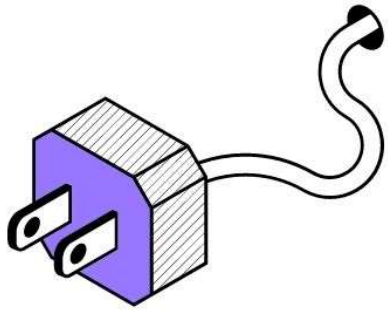
- **public class** Circulo **extends** ObjetoGeometrico

Tenga en cuenta los siguientes puntos con respecto a la herencia

- Los campos de datos privados en una superclase no son accesibles fuera de la clase

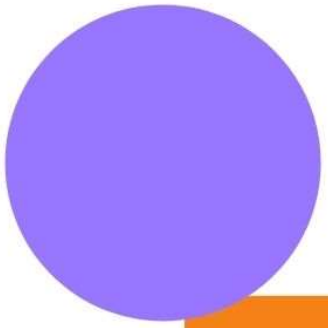
- La herencia se utiliza para modelar la relación **es-un**

Polimorfismo



Conceptos clave

- Etimológicamente significa muchas formas.
- En programación orientada a objetos, polimorfismo es la capacidad que tienen los objetos de una clase en ofrecer respuesta distinta e independiente en función de los parámetros (diferentes implementaciones) utilizados durante su invocación.



Sobre escritura de un método

- Para sobre escribir un método, el método debe definirse en la subclase utilizando la misma firma y el mismo tipo de retorno que en su superclase.

- Una subclase hereda métodos de una superclase. A veces es necesario que la subclase modifique la implementación de un método definido

Tenga en cuenta los siguientes puntos con respecto a la sobre escritura

- Un método de instancia puede sobre escribirse solo si es accesible. Por lo tanto, un método privado no puede sobre escribirse porque no es accesible fuera de su propia clase.

- Al igual que un método de instancia, un método estático se puede heredar. Sin embargo, un método estático no se puede sobre escribir. Si un método estático definido en la superclase se redefine en una subclase, el método definido en la superclase se oculta.