

# Sobrescritura de métodos

# Concepto

➤ Cuando una clase hereda un método de otra puede sobrescribirlo, lo que significa que vuelve a definirlo en la nueva clase:

```
class Clase1{  
    public void test(){  
        System.out.println("uno");  
    }  
}  
class Clase2 extends Clase1{  
    //el método vuelve a definirse  
    public void test(){  
        System.out.println("dos");  
    }  
}
```



```
Clase2 c=new Clase2();  
c.test(); //imprime dos
```

# Anotación @Override

➤ Indica al compilador que se está intentando sobrescribir un método. Su uso no es obligatorio, pero sí conveniente:

```
class Clase1{  
    public void test(){  
        System.out.println("uno");  
    }  
}  
class Clase2 extends Clase1{  
    @Override  
    public void Test(){  
        System.out.println("dos");  
    }  
}
```

Error  
compilación

```
class Clase1{  
    public void test(){  
        System.out.println("uno");  
    }  
}  
class Clase2 extends Clase1{  
    public void Test(){  
        System.out.println("dos");  
    }  
}
```

No error compilación,  
el compilador no  
sabe que intentamos  
sobrescribir

# Reglas sobrescritura

➤ A la hora de sobrescribir un método, se deben seguir las siguientes reglas:

- El nombre y lista de parámetros debe ser idéntico
- El ámbito debe ser igual o menos restrictivo
- El tipo de devolución debe ser igual o un subtipo del original
- La nueva versión del método no debe propagar excepciones que no estén definidas en el original (esta restricción NO afecta a las excepciones Runtime)

# Sobrescritura vs sobrecarga

- Es común confundir ambas características cuando hay herencia entre clases.
- El siguiente ejemplo sería un caso de sobrecarga, no sobrescritura:

```
class Clase1{  
    public void test(){  
    }  
}  
class Clase2 extends Clase1{  
    //el método incluye un parámetro  
    public void test(int s){  
  
    }  
}
```

La nueva clase  
dispone ahora de  
dos métodos test

# Ejemplos sobrescritura correcta

```
class Clase1{  
    public Object test(){ }  
}  
class Clase2 extends Clase1{  
    @Override  
    public String test(){ }  
}
```

Tipo devolución  
subclase de  
Object

Ámbito  
superior a  
(default)

```
class Clase1{  
    void test(){ }  
}  
class Clase2 extends Clase1{  
    @Override  
    public void test(){ }  
}
```

```
class Clase1{  
    void test() throws IOException{ }  
}  
class Clase2 extends Clase1{  
    @Override  
    public void test() throws FileNotFoundException{ }  
}
```

Subtipo de  
IOException

# Ejemplos sobrescritura incorrecta

```
class Clase1{  
    public void test(){ }  
}  
class Clase2 extends Clase1{  
    @Override  
    public String test(){ } //error compilación  
}
```

El tipo  
devolución  
debería ser void

Ámbito inferior  
a public

```
class Clase1{  
    public void test(){ }  
}  
class Clase2 extends Clase1{  
    @Override  
    void test(){ } //error compilación.  
}
```

```
class Clase1{  
    void test(){ }  
}  
class Clase2 extends Clase1{  
    @Override  
    public void test() throws SQLException{ } //error compilación.  
}
```

Excepción no declarada  
en la superclase