## SUPER

Nos permite acceder desde una subclase a su superclase (sus métodos).

Por ejemplo:

### ACCESO A LA SUPERCLASE

- Si uno de nuestros métodos sobreescribe un método de la clase base, podemos invocar este a través de la palabra **super**.

```
// overrides printMethod in Superclass
public void printMethod() {
   super.printMethod();
   System.out.println("Printed in Subclass");
}
```

Una subclase puede complementar la funcionalidad del método que existe en super, implementando un método con el mismo nombre, invocar al del super y añadir algunas sentencias adicionales.

Esto lo hemos hecho ya, con los constructores de la subclase:

### CONSTRUCTORES Y SUPER

- Un constructor de una subclase puede usar **super** para invocar a un constructor de su clase base.
- Si una subclase no lo invoca, la JVM lo hace por él. La clase base debe tener entonces un constructor sin parámetros.

```
public Empleado(…) {
     super(nombre, puesto, direccion, telefono, nSS);
     this.sueldo = sueldo;
     this.impuestos = impuestos;
}
```

Mira este ejemplo, tengo el método imprimir en la super y en la sub clase, de forma que la subclase añade una sentencia:

```java
public class ClaseBase {

    public void imprimir() {
        System.out.println("Saludo desde la clase base");
    }

}
```

```java
public class ClaseDerivada extends ClaseBase {

    public void imprimir() {
        super.imprimir();
        System.out.println("Saludo desde la clase derivada");
    }
}
```

Esta clase que extiende de ClaseBase, implementa un método con el mismo nombre en el que llama al del super y añade una sentencia más.

Si en el método main instancias un objeto de cada clase, prueba sus métodos:

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

    ClaseBase base = new ClaseBase();
    ClaseDerivada derivada = new ClaseDerivada();

    base.imprimir();
    System.out.println("");
    derivada.imprimir();


}
```

Cuidado con los constructores, si añadís un constructor en claseBase que reciba un String, puede generar errores en las subclases, sino tenéis constructor vacío en la claseBase.