

Métodos default en Interfaces. Añadir métodos con cuerpo en las interfaces: Hasta ahora, las interfaces solo podían contener cabecera de métodos abstractos y constantes. Ejemplo:

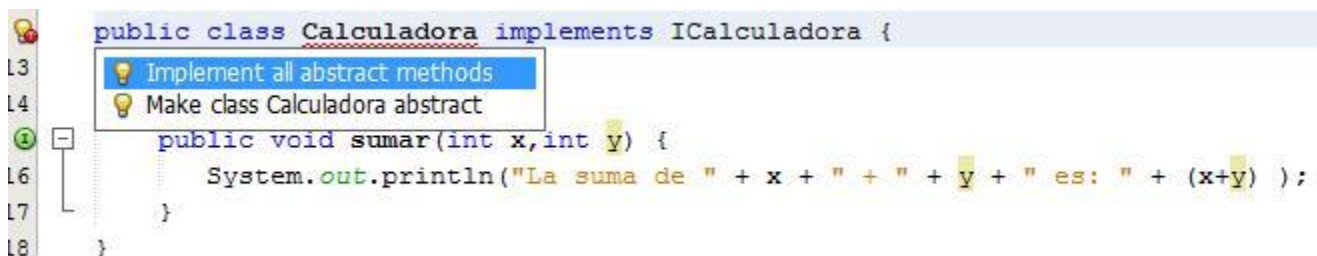
```
public interface ICalculadora {  
    public void sumar(int x,int y);  
}
```

Una clase que implementa dicha interfaz debe contener como mínimo la implementación de los métodos que han sido declarados en la interfaz. Todo tipo de calculadora que implemente ICalculadora podrá realizar todas las operaciones que se indiquen en esta última.

```
public class Calculadora implements ICalculadora {  
  
    @Override  
    public void sumar(int x,int y) {  
        System.out.println("La suma de " + x + " + " + " + y + " es: " + (x+y) );  
    }  
}
```

Por tanto, si en la interfaz se añade posteriormente un método abstracto, en la clase Java que implementa a dicha interfaz encontraríamos un **error de compilación "NECESITA IMPLEMENTAR EN LA CLASE EL NUEVO MÉTODO CREADO EN LA INTERFAZ"**

```
public interface ICalculadora {  
    public void sumar(int x,int y);  
    public int restar (int x, int y);  
}
```



```
public class Calculadora implements ICalculadora {  
    public void sumar(int x,int y) {  
        System.out.println("La suma de " + x + " + " + " + y + " es: " + (x+y) );  
    }  
}
```

Implement all abstract methods
Make class Calculadora abstract

Este error se soluciona, añadiendo a la **clase la implementación** del método "restar":

```
public class Calculadora implements ICalculadora {

    @Override
    public void sumar(int x,int y) {
        System.out.println("La suma de " + x + " + " + y + " es: " + (x+y) );
    }

    @Override
    public int restar (int x,int y){
        return x-y;
    }
}
```

¿Qué hay de nuevo ahora en Java 8?

Podremos encontrarnos con un nuevo elemento en las interfaces: **métodos por defecto**. Estos métodos se caracterizan porque ya en la interfaz podemos declararles un comportamiento por defecto.

```
public interface ICalculadora {
    public void sumar(int x,int y);
    public int restar (int x, int y);

    public default int multiplicar (int x,int y){
        return x*y;
    }
}
```

Estos métodos no se tienen que declarar en la clase que implementa a la interfaz, directamente podrán ser utilizados. Realizamos un método principal para probar las distintas operaciones que podremos realizar desde una Calculadora y vemos cómo podemos hacer uso del método multiplicar desde un objeto Calculadora sin que dicho método esté implementado en la clase Calculadora (esto se debe a que dicho método se ha declarado como método default en la interfaz ICalculadora):


```
15 public class ProyectoJava8 {
16
17     public static void main(String[] args) {
18         Calculadora c = new Calculadora();
19         c.sumar(4, 3);
20         System.out.println("La multiplicación es " +c.multiplicar(2, 3));
21     }
22 }
```

Output - ProyectoJava8 (run) %

```
run:
La suma de 4 + 3 es: 7
La multiplicación es 12
```

Tras ver el ejemplo, cabe destacar que en una interfaz podremos añadir **nuevos comportamientos** por defecto (**métodos default**) que no tendrán por qué ser tratados en todas las clases que implementen a dicha interfaz, pero sí **podrán ser utilizados en todas ellas**.