Estructura y definición de los métodos en Java

David Corzo
Anesveth Maatens
Ian Jenatz
Adriana Mundo

2019-Oct-29 18:28:23

1. Definiciones preliminares

Recordemos que un método se define dentro de una clase y se declara bajo un nombre, posteriormente en la ejecución de nuestro programa lo podemos instanciar o invocar simplemente por su nombre. Los métodos definidos en java pueden ser de dos tipos, pueden ser de tipo "void" que simplemente significa que no devolverán ningún valor, y los métodos que devuelven un valor que hay una variedad de formas que se pueden definir.

2. Declaración de métodos en Java

Observar el siguiente código:

```
public class ExplainMethods {
    // Método que retorna un valor
    static int value_return( /* Los parámetros van aquí */ ) {
        // Hagámos el cálculo de un número al cuadrado...
        int numero = 8;
        int square = numero * numero; // Respuesta 64
        return square;
    }

    // Este método no imprimirá nada en pantalla
    // Pero sí retorna un valor
    // Ahora un void
    static void no_value_return(/* Parámetros van aquí */) {
        // Hagamos un cálculo de un número
        int numero = 5;
        System.out.println(numero + 8); // Respuesta 13
    }

    // El punto de entrada a mi programa es el método main
    public static void main(String[] args) {
        System.out.println("Return method");
        // Como el método devuelve algo no imprimirá nada, veámoslo:
        System.out.println(value_return());
        System.out.println("Void method");
        no_value_return();
    }
```

```
}
```

Podemos observar un ejemplo de los dos tipo de métodos el de void con el no_value_return y el de int con value_return. Los métodos se utilizan para la realización de acciones dentro del programa y se definen de alguna de las dos maneras ejemplificadas anteriormente.

3. Estructura y declaración de métodos en Java

Posteriormente a haber definido una clase (en el ejemplo anterior la clase es nombrada "ExplainMethods") dentro de ella se declaran los métodos a implementar, la estructura de los métodos en java es primero elegir que va a retornar la función, si no va a retornar algo usar el "void declaration" de lo contrario uno debe declarar el tipo de dato que va a retornar a priori a la implentación del código, en el ejemplo anterior el dato de retorno de "no_value_return" lo que retornaba era un entero por lo tanto se declaró el método bajo el tipo de dato int.

3.1. Void

El método void (el que no devuelve nada) se declara con la siguiente sintaxis:

```
static void <<nombre_del_método>>( <<parámetros>> ){ <<acciones>> ;}
```

En «acciones» van todas las cosas que se deben ejecutar en el método y cada línea de código por supuesto terminando con ";".

3.2. Métodos por tipo de dato

El método que se implementa a partir de un tipo de dato se implementa con la siguiente sintaxis:

```
static <<tipo_de_dato>> <<nombre_del_método>> { <<acciones>>; return <<valor en el
↪  tipo de dato declarado en el método>>; }
```

Posterior a las «acciones» se retorna un valor de cualquier tipo de dato ya sea float, char o int con «return valor a retornar».

4. Modificadores de acceso

Una de las cosas que hay que tomar en cuenta en la estructura de los métodos son los modificadores de acceso, estos son exactamente el nombre la a entender, modifican el acceso que uno tiene a poder llamar o cambiar los métodos en cuestión. La gramática de dichos modificadores de acceso comprende cuatro palabras clave en java, entre estas están public, private, protected & un tipo que no tiene palabra asociada que se le delega el nombre de "default". Dichos modificadores privan, protegen o hacen público el acceso por medio de otras clases a variables y métodos de una clase en específico.

1. **Public**: Da a dicho método la característica que pueden ser accesados globalmente desde todas las clases.

2. **Private**: Estos solo pueden ser accesados por la misma clase en las que son declarados, por lo mismo priva a las demás clases de este método. Este solo puede ser accesado desde la misma clase.

3. **Protected**: Este modificador excluye el acceso a sólo poder ser accedidos por el mismo paquete que declara a la clase, en cierta manera es público solo en el paquete donde está declarado. Este puede ser accesados desde la misma clase, desde otra clase del mismo paquete y desde una subclase de otro paquete.

4. **Default**: Si no se declara ninguno de los modificadores anteriores se dará el modificador default por java, este hace que el método pueda ser accesado por las clases que están declaradas en el mismo paquete, estas tienen menos accesos que las protected, pueden ser accesados por la misma clase y por otra clase del mismo paquete únicamente.

Es importante tomar en cuenta dichos modificadores de acceso a la hora de llamar métodos en java dado a que si son declarados con un tipo que priva su uso el programa dará error.

4.1. Gramática de modificadores

```
<modificador de acceso> class <tipo de dato o void> <nombre del método>
↪  {<<métodos>>; <<main>>;}
Ejemplo...
public class int numero_a_calcular() {<<acciones>>;}
```

4.2. Paréntesis

El keyword "static" se refiere a la declaración de miembros de la clase, el método "main", se debe declarar como `public static void main() {}` por esta razón.

5. En suma...

Los la estructura de los métodos en java sigue la sintaxis presentada anteriormente y posterior a haber decidido qué tipo de método se va a implementar si uno utiliza void el método implementado no devolverá nada, de lo contrario hay que emplear el keyword "return «valor a retornar»". Todo esto definido dentro de una clase con la sintaxis "public class «nombre de la clase» «métodos»; «main»;".