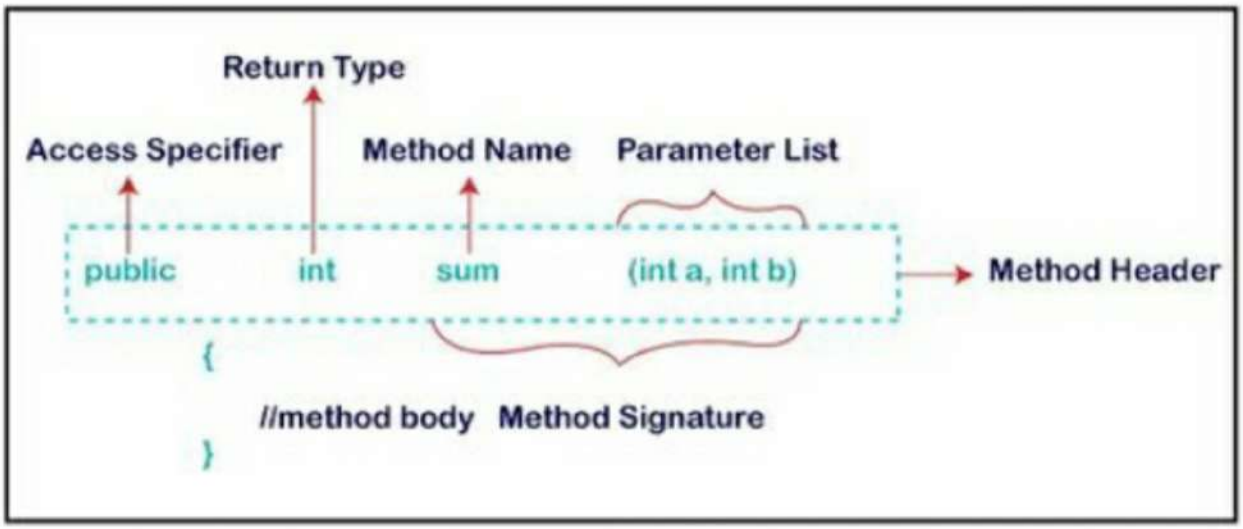
JAVA

FUNCTIONS / METHOD

Jayesh Deshmukh
@jayeshdeshmukh

A **method** is a block of code that performs a specific task.

Method Declaration



1. Modifier: It defines the access type of the method i.e. from where it can be accessed in your application. In Java, there 4 types of access specifiers.

- **public:** It is accessible in all classes in your application.
- **protected:** It is accessible within the class in which it is defined and in its subclass.
- **private:** It is accessible only within the class in which it is defined.

Jayesh Deshmukh
@jayeshdeshmukh

- **default:** It is declared/defined without using any modifier. It is accessible within the same class and package within which its class is defined.

2. The return type: The data type of the value returned by the method or void if does not return a value.

3. Method Name: the rules for field names apply to method names as well, but the convention is a little different.

4. Parameter list: Comma-separated list of the input parameters is defined, preceded with their data type, within the enclosed parenthesis. If there are no parameters, you must use empty parentheses ( ).

5. Method Signature: Every method has a method signature. It is a part of the method declaration. It includes the method name and parameter list.

6. Method body: it is enclosed between braces. The code you need to be executed to perform your intended operations.

Jayesh Deshmukh
@jayeshdeshmukh

Types of Methods in Java

- **User-defined Methods:** We can create our own method based on our requirements.
- **Predefined Method:** These are built-in methods in Java that are available to use.

Method Calling

we have declared a method named **addNumbers( ).** Now, to use the method, we need to call it.

```
class Main {
  // create a method
  public int addNumbers(int a, int b) {
    int sum = a + b;
    return sum;  // return value
  }
  public static void main(String[] args) {
    int num1 = 25;
    int num2 = 15;
    Main obj = new Main();  // create an object of Main
    int result = obj.addNumbers(num1, num2);  // calling method
    System.out.println("Sum is: " + result);
  }
}
```

Jayesh Deshmukh
@jayeshdeshmukh

Method Return Type

A Java method may or may not return a value to the function call. We use the **return statement** to return any value.

```java
class Main {
// create a method
  public static int square(int num) {
    // return statement
    return num * num;
  }
  public static void main(String[] args) {
    int result;
    // call the method
    // store returned value to result
    result = square(10);
    System.out.println("Squared value of 10 is: " + result);
  }
}
```

Method Parameters

```java
// method with two parameters
int addNumbers(int a, int b) {
  // code
}

// method with no parameter
int addNumbers( ){
  // code
}
```

Jayesh Deshmukh
@jayeshdeshmukh

```java
class Main {
  // method with no parameter
  public void display1() {
    System.out.println("Method without parameter");
  }
  // method with single parameter
  public void display2(int a) {
    System.out.println("Method with a single parameter: " + a);
  }
  public static void main(String[] args) {
    Main obj = new Main();  // create an object of Main
    obj.display1();  // calling method with no parameter
    obj.display2(24);  // calling method with the single parameter
  }
}
```

Predefined Method

Predefined methods are the method that is already defined in the Java class libraries is known as predefined methods. It is also known as the **standard library method** or **built-in method**. We can directly use these methods just by calling them in the program at any point. Some pre-defined methods are **length(), equals(), compareTo(), sqrt(),** etc.

Jayesh Deshmukh
@jayeshdeshmukh

```java
public class Main {
  public static void main(String[] args) {
    // using the sqrt() method
    System.out.print("Square root of 4 is: " + Math.sqrt(4));
  }
}
```

What are the advantages of using methods?

The main advantage is **code reusability.** We can write a method once, and use it multiple times. We do not have to rewrite the entire code each time. Think of it as, "write once, reuse multiple times".

```java
public class Main {
  // method defined
  private static int getSquare(int x){
    return x * x;
  }
  public static void main(String[] args) {
    for (int i = 1; i <= 5; i++) {
      int result = getSquare(i);  // method call
      System.out.println("Square of " + i + " is: " + result);
    }
  }
}
```

Jayesh Deshmukh
@jayeshdeshmukh

Jayesh Deshmukh

Follow

Like Comment Share Save