Declaring a Java Method

The syntax to declare a method is:

```
returnType methodName() {
  // method body
}
```

Here,

- returnType - It specifies what type of value a method returns For example if a method has an `int` return type then it returns an integer value.
  If the method does not return a value, its return type is `void`.
- methodName - It is an identifier that is used to refer to the particular method in a program.
- method body - It includes the programming statements that are used to perform some tasks. The method body is enclosed inside the curly braces `{ }`.

For example,

```
int addNumbers() {
// code
}
```

In the above example, the name of the method is `adddNumbers()`. And, the return type is `int`. We will learn more about return types later in this tutorial.

This is the simple syntax of declaring a method. However, the complete syntax of declaring a method is

```
modifier static returnType nameOfMethod (parameter1, parameter2, ...) {
  // method body
}
```

modifier - It defines access types whether the method is public, private, and so on.

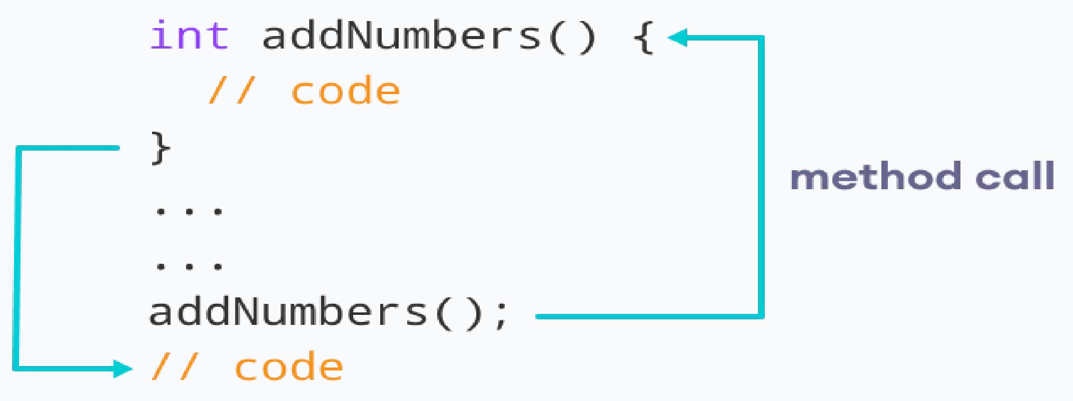
static - If we use the `static` keyword, it can be accessed without creating objects.

- parameter1/parameter2 - These are values passed to a method. We can pass any number of arguments to a method.

Calling a Method in Java

In the above example, we have declared a method named addNumbers(). Now, to use the method, we need to call it.
Here's how we can call the addNumbers() method.

```
// calls the method
addNumbers();
```



Example 1: Java Methods

```java
class Main {

  // create a method
  public int addNumbers(int a, int b) {
    int sum = a + b;
    // return value
    return sum;
  }

  public static void main(String[] args) {

    int num1 = 25;
    int num2 = 15;

    // create an object of Main
    Main obj = new Main();
    // calling method
    int result = obj.addNumbers(num1, num2);
    System.out.println("Sum is: " + result);
  }
```

```
    }
```

Output

```
Sum is: 40
```

In the above example, we have created a method named `addNumbers()`. The method takes two parameters `a` and `b`. Notice the line,

```
int result = obj.addNumbers(num1, num2);
```

Here, we have called the method by passing two arguments `num1` and `num2`. Since the method is returning some value, we have stored the value in the `result` variable.

Java 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. For example,

```java
int addNumbers() {
...
return sum;
}
```

Here, we are returning the variable `sum`. Since the return type of the function is `int`. The sum variable should be of `int` type. Otherwise, it will generate an error.

Example 2: Method Return Type

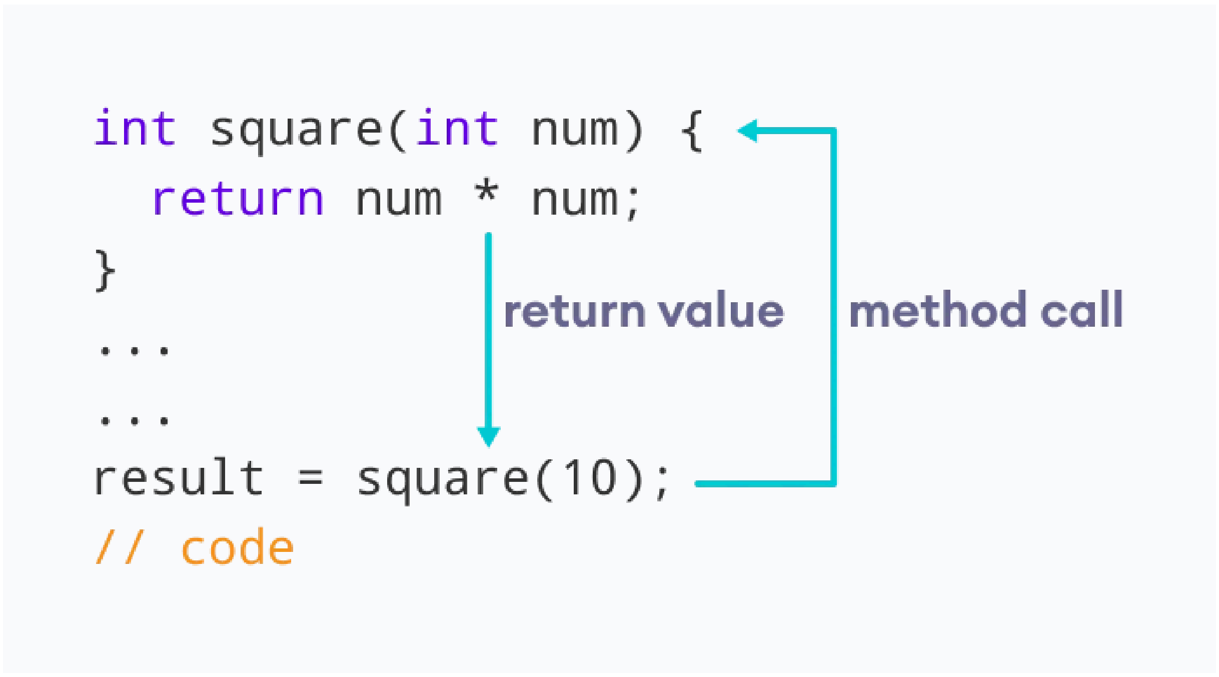
```java
class Main {

// 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);
  }
}
```

Output:

```
Squared value of 10 is: 100
```

In the above program, we have created a method named `square()`. The method takes a number as its parameter and returns the square of the number.

Here, we have mentioned the return type of the method as `int`. Hence, the method should always return an integer value.


```
int square(int num) {
  return num * num;
}
...
...
result = square(10);
// code
```

Method Parameters in Java

A method parameter is a value accepted by the method. As mentioned earlier, a method can also have any number of parameters. For example,

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

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

If a method is created with parameters, we need to pass the corresponding values while calling the method. For example,

// calling the method with two parameters
addNumbers(25, 15);

// calling the method with no parameters
addNumbers()

Example 3: Method Parameters

```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) {

    // create an object of Main
    Main obj = new Main();

    // calling method with no parameter
    obj.display1();

    // calling method with the single parameter
    obj.display2(24);
  }
}
```

Output

```
Method without parameter
```

Method with a single parameter: 24