method in Java:

```java
returnType methodName(parameters) {
    // Method body
    // Code to perform the task
    // ...
    return returnValue; // (if the method has a return type)
}
```

Here's an example of defining and using a method:

```java
public class ExampleMethods {
    // Method to calculate the sum of two integers
    public static int calculateSum(int num1, int num2) {
        int sum = num1 + num2;
        return sum; // Return the calculated sum
    }

    public static void main(String[] args) {
        int a = 5;
        int b = 7;
        int result = calculateSum(a, b); // Call the method and store the result
        System.out.println("The sum of " + a + " and " + b + " is: " + result);
    }
}
```

In this example:

- The method `calculateSum` takes two integer parameters `num1` and `num2`.
- It calculates the sum of the two numbers and returns the result.
- The `main` method calls `calculateSum` with values `5` and `7`, then prints the result.

Method Components:

1. **Return Type**:
   The `returnType` specifies the type of value the method will return. It can be `void` (no return value) or any other data type.

2. **Method Name**:
   The `methodName` is a meaningful name that describes the purpose of the method. It follows Java naming conventions.

3. **Parameters**:
   The `parameters` are variables declared inside parentheses and used as inputs to the method. You can have zero or more parameters.

4. **Method Body**:
   The method body contains the code that performs the task. It's enclosed in curly braces `{}`.

5. **Return Statement** (optional):
   If the method has a return type other than `void`, you use the `return` statement to specify the value to be returned.

Using Methods for Reusability:

One of the primary benefits of methods is code reusability. You can define a method once and call it from various parts of your program.

```java
public class ExampleMethods {
    public static int calculateSquare(int num) {
        return num * num;
    }

    public static void main(String[] args) {
        int x = 5;
        int squareX = calculateSquare(x);
        System.out.println("The square of " + x + " is: " + squareX);

        int y = 8;
        int squareY = calculateSquare(y);
        System.out.println("The square of " + y + " is: " + squareY);
    }
}
```

In this example, the `calculateSquare` method is defined once but can be used multiple times to calculate the square of different numbers.

Methods are essential for building maintainable and organized programs. They allow you to encapsulate functionality, promote reusability, and make your code easier to understand and maintain.