Java Fundamentals: Arrays

This guide explains **Arrays**, including **One-dimensional arrays** and **Two-dimensional arrays** in Java. Arrays are a collection of elements, all of the same type, stored in contiguous memory locations.

---

1. Arrays in Java

An **array** is a container object that holds a fixed number of values of a single data type. The elements of an array are indexed, starting from 0.

Syntax to declare an array:

```
dataType[] arrayName = new dataType[arraySize];
```

- **dataType**: The type of elements to be stored in the array (e.g., `int`, `String`).
- **arrayName**: The name of the array.
- **arraySize**: The number of elements the array will hold.

Example of Declaring an Array:

```
int[] numbers = new int[5];  // Array of 5 integers
```

---

2. One-dimensional Arrays

A **one-dimensional array** is a list of elements that are accessed via a single index.

Syntax for One-dimensional Array:

```
dataType[] arrayName = new dataType[size];
```

Or, if the array is initialized with values:

```
dataType[] arrayName = {value1, value2, ..., valueN};
```

Example of One-dimensional Array:

```java
class OneDimensionalArray {
    public static void main(String[] args) {
        // Declare and initialize an array
        int[] numbers = {10, 20, 30, 40, 50};

        // Accessing elements of the array
        System.out.println("First element: " + numbers[0]);  // Output: 10
        System.out.println("Second element: " + numbers[1]);  // Output: 20
        System.out.println("Last element: " + numbers[4]);  // Output: 50

        // Iterating through the array
        System.out.println("\nArray elements:");
        for (int i = 0; i < numbers.length; i++) {
            System.out.println("Element at index " + i + ": " + numbers[i]);
        }
    }
}
```

Output:

```
First element: 10
Second element: 20
Last element: 50

Array elements:
Element at index 0: 10
Element at index 1: 20
Element at index 2: 30
Element at index 3: 40
Element at index 4: 50
```

---

3. Two-dimensional Arrays

A **two-dimensional array** is essentially an array of arrays. It can be visualized as a matrix with rows and columns.

Syntax for Two-dimensional Array:

```java
dataType[][] arrayName = new dataType[rows][columns];
```

Or, if the array is initialized with values:

```java
dataType[][] arrayName = {{value1, value2, ..., valueN}, {value1, value2, ..., valueM}};
```

Example of Two-dimensional Array:

```
class TwoDimensionalArray {
    public static void main(String[] args) {
        // Declare and initialize a 2D array
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        // Accessing elements of the 2D array
        System.out.println("Element at [0][0]: " + matrix[0][0]);  // Output: 1
        System.out.println("Element at [2][2]: " + matrix[2][2]);  // Output: 9

        // Iterating through the 2D array
        System.out.println("\nArray elements:");
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[i].length; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Output:

```
Element at [0][0]: 1
Element at [2][2]: 9

Array elements:
1 2 3
4 5 6
7 8 9
```

---

4. Accessing and Modifying Array Elements

You can access and modify elements of arrays using their indices. Array indices start from **0**.

Example - Modifying Elements:

```
class ModifyArray {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, 5};
```

```
        // Modify an element at index 2
        numbers[2] = 10;

        // Print the modified array
        for (int i = 0; i < numbers.length; i++) {
            System.out.println("Element at index " + i + ": " + numbers[i]);
        }
    }
}
```

Output:

```
Element at index 0: 1
Element at index 1: 2
Element at index 2: 10
Element at index 3: 4
Element at index 4: 5
```

---

5. Multi-dimensional Arrays (N-dimensional Arrays)

Java allows creating arrays with more than two dimensions, though two-dimensional arrays are the most common.

Example of a Three-dimensional Array:

```
class ThreeDimensionalArray {
    public static void main(String[] args) {
        int[][][] cube = {
            {{1, 2}, {3, 4}},
            {{5, 6}, {7, 8}},
            {{9, 10}, {11, 12}}
        };

        // Accessing an element in a 3D array
        System.out.println("Element at [1][0][1]: " + cube[1][0][1]);  // Output: 6
    }
}
```

Output:

```
Element at [1][0][1]: 6
```

Summary

- **One-dimensional arrays** store a list of values, accessed via a single index.
- **Two-dimensional arrays** store values in a matrix format (rows and columns).
- You can declare and initialize arrays with or without values.
- Arrays are indexed from **0** in Java.
- **Multi-dimensional arrays** allow the creation of more complex data structures like 3D arrays, and beyond.
