Java Array

Java Array

In this tutorial, we will learn what an array is in Java, how to initialize it, how to access the elements of array, how to modify the elements, etc.

What is an Array in Java?

In Java programming language, array is a collection of elements that belong to similar datatype. An array is stored in memory in continuous locations.

How to declare an Array in Java?

Following is the syntax to declare an Array in Java.

```
datatype arrayName[];
```

See the placement of square brackets, right after the array name. Or you can also place the square brackets right after datatype.

```
datatype[] arrayName;
```

You can use any of these two notations.

So, to define an integer array, you would write a statement as shown below.

```
int numbers[];
//or
int[] numbers;
```

When you declare an array, you are only reserving the variable name and telling the compiler that you are going to use this variable name to store elements of this particular datatype.

How to initialize an Array in Java?

To initialize an array, you can assign the array variable with new array of specific size as shown below.

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

You have to mention the size of array during initialization. This will create an array in memory, with all elements initialized to their corresponding static default value.

For example, to initialize an integer array, see the following program.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int numbers[];
        numbers = new int[10];
    }
}
```

We have first declared the variable and then initialized it. Of course, we have declared and initialized the array in two different statements. But you can combine the declaration and initialization, to form the definition of array, as shown below.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int numbers[] = new int[10];
    }
}
```

In the above example, we have created an integer array named **numbers**, and initialized it to an integer array of size **10** with default values. The default value for an integer is **0**.

You can also assign elements directly to the array when declaring it.

In the following example, we have declared and initialized array with elements.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10};
    }
}
```

Now **numbers** is an integer array with size of 5, because there are five elements in the array we assigned.

The size of array is fixed. You can only modify the existing elements of the array. But you cannot remove elements or add elements to the array.

How to access Array Elements?

Now that we have initialized an array with elements, how can we access them? Remember, we mentioned during introduction that an array is stored in continuous memory locations. And these elements can be accessed using index. index is the position of the element from starting of the array. The first element has an index of 0, second element has an index of 1, third element has an index of 2 and so on.

Following is the syntax to access an element of array.

```
arrayName[index]
```

The above statement can be used either to read an element at given index, or set an element at given index. The read or write operation depends on which side of assignment operator you are placing this.

For example, in the following program, we are reading the element of array **nums** at index **4**.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10, 12, 14, 16};
        int n = numbers[4];
        System.out.println(n);
    }
}
```

Output

```

```

And in the following example, we are updating the element of array at index **4**.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10, 12, 14, 16};
        numbers[4] = 25;
        System.out.println(numbers[4]);
    }
}
```

```
}
```

Output

```

```

How to iterate over array elements?

From all the above discussion, we now know that array is a collection of elements stored in a continuous memory. And we can use index to access the elements.

Now, how do we access elements of an array one by one in a loop? Well! Java has looping statements like while loop, for loop and advanced for loop. We can use any of these looping statements to iterate over elements of a Java Array.

In the following example, we have initialize a Java Array, and iterate over elements of Array using Java While Loop.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10};
        int index = 0;
        while (index < numbers.length) {
            System.out.println(numbers[index]);
            index++;
        }
    }
}
```

Output

```
2
4
6
8
10
```

In the following example, we have initialize a Java Array, and iterate over elements of Array using Java For Loop.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10};
        for (int index=0; index < numbers.length; index++) {
```

```
            System.out.println(numbers[index]);
        }
    }
}
```

Output

```
2
4
6
8
10
```

And in the following example, we will use advanced for loop, to iterate over elements of array.

Java Program

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10};
        for (int n: numbers) {
            System.out.println(n);
        }
    }
}
```

Output

```
2
4
6
8
10
```

More Tutorials about Java Arrays

Following are some of the in-depth tutorials for some niche concepts of Java Arrays.

- How to initialize Array in Java?
- Java Array of Integers
- Java Array of Strings
- Java Array – While Loop
- Java Array – For Loop
- Java Array – ForEach
- Concatenate Arrays in Java
- Array of Objects in Java
- Array of Arrays in Java
- Find Sum of Elements of Java Array
- Find Average of Elements in Java Array

Conclusion

In this Java Tutorial, we learned everything we need to know, to start using Java Arrays in your program.

Java Tutorial

- Java Tutorial
- Java Introduction
- Java Installation
- IDEs for Java Development
- Java HelloWorld Program
- Java Program Structure
- Java Datatypes
- Java Variable Types
- Java Access Modifiers
- Java Operators
- Java Decision Making
- Java Loops
- **Java Array**
- Java OOPs
- Java String
- Java Exception Handling
- Java File Operations
- Java Date & Time
- Java MySQL
- Java Random
- Java Math