Java Arrays

Press Space for next page →

Agenda

1. Java Arrays
2. Arrays Memory Allocation
3. Types of Arrays
4. Declaring an Array
5. Code Example of Java Array
6. Code Example of Returning Array from Method
7. Java ArrayIndexOutOfBoundsException
8. Code Example of Multi-Dimensional Array

Java Arrays

> An array is a collection of similar type of elements that have a contiguous memory location.

Learn more about Java Arrays

Java Arrays

> An array is a collection of similar type of elements that have a contiguous memory location.

- An array is a collection of similar types of data

Learn more about Java Arrays

Java Arrays

> An array is a collection of similar type of elements that have a contiguous memory location.

- An array is a collection of similar types of data
- It is a fixed-size sequential collection of elements of the same type

Learn more about Java Arrays

Java Arrays

> An array is a collection of similar type of elements that have a contiguous memory location.

- An array is a collection of similar types of data
- It is a fixed-size sequential collection of elements of the same type
- It is a data structure where we store similar elements

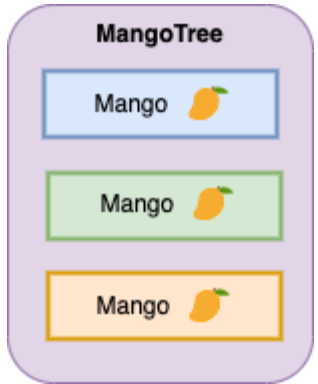
Learn more about Java Arrays

MangoTree

MangoTree

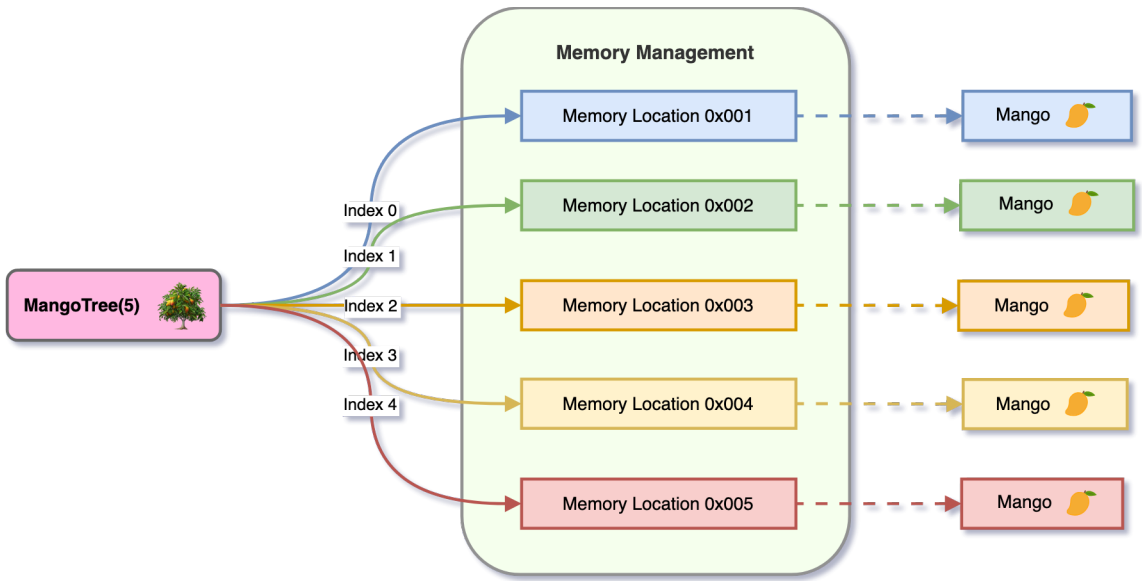
MangoTree

MangoTree



Arrays Memory Allocation

> Memory allocation for arrays is done in a continuous block of memory.



Types of Arrays

- Single Dimensional Array
- Multi-Dimensional Array

Declaring an Array

Instantiation of an Array in Java

Declaring an Array

- `dataType[] arr;`

Instantiation of an Array in Java

Declaring an Array

- `dataType[] arr;`
- `dataType arr[];`

Instantiation of an Array in Java

Declaring an Array

- `dataType[] arr;`
- `dataType arr[];`
- `dataType arr[];`

Instantiation of an Array in Java

Declaring an Array

- `dataType[] arr;`
- `dataType arr[];`
- `dataType arr[];`

Suggested: dataType[] arr;

Instantiation of an Array in Java

Declaring an Array

- `dataType[] arr;`
- `dataType arr[];`
- `dataType arr[];`

Suggested: dataType[] arr;

Instantiation of an Array in Java

- `dataType[] arr = new datatype[size];`

Declaring an Array

- `dataType[] arr;`
- `dataType arr[];`
- `dataType arr[];`

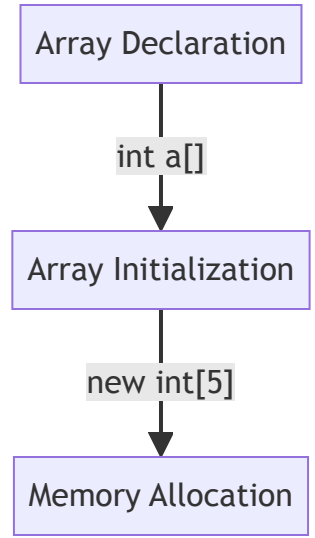
Suggested: dataType[] arr;

Instantiation of an Array in Java

- `dataType[] arr = new datatype[size];`
- `int[] intArray = {48,5,4,5};` **declaration, instantiation and initialization**

Code Example of Java Array

```
class JavaArrayExample {
    public static void main(String[] args) {
        int[] intArray = new int[5];
        intArray[0] = 10;
        intArray[1] = 20;
        intArray[2] = 30;
        intArray[3] = 40;
        intArray[4] = 50;
        for (int i = 0; i < intArray.length; i++) {
            System.out.println(intArray[i]);
        }
    }
}
```



Code Example of Returning Array from Method

```
class JavaArrayExample {
    public static void main(String[] args) {
        int[] intArray = getArray();
        for (int i = 0; i < intArray.length; i++) {
            System.out.println(intArray[i]);
        }
    }

    public static int[] getArray() {
        int[] intArray = new int[5];
        intArray[0] = 10;
        intArray[1] = 20;
        intArray[2] = 30;
        intArray[3] = 40;
        intArray[4] = 50;
        return intArray;
    }
}
```

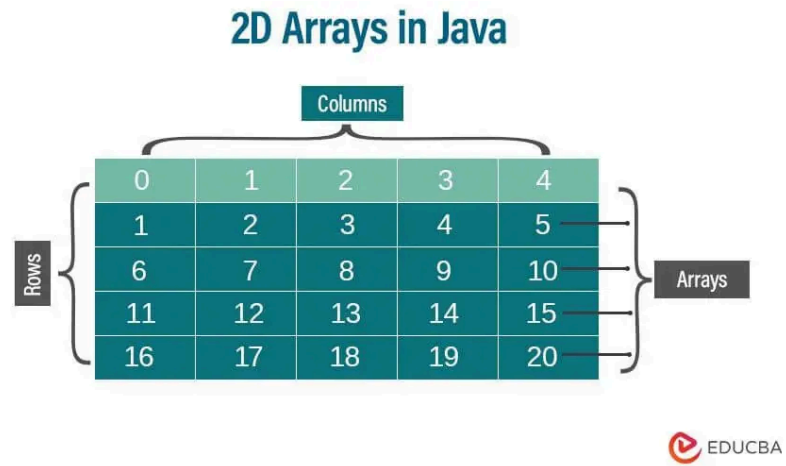
Java ArrayIndexOutOfBoundsException

An ArrayIndexOutOfBoundsException is thrown when you try to access an array element with an index that is either less than zero or greater than or equal to the length of the array.

```
class JavaArrayExample {
    public static void main(String[] args) {
        int[] intArray = new int[5];
        intArray[5] = 10;
    }
}
```

Code Example of Multi-Dimensional Array

```
class JavaArrayExample {
    public static void main(String[] args) {
        int[][] intArray = new int[2][3];
        intArray[0][0] = 10;
        intArray[0][1] = 20;
        intArray[0][2] = 30;
        intArray[1][0] = 40;
        intArray[1][1] = 50;
        intArray[1][2] = 60;
        for (int i = 0; i < intArray.length; i++) {
            for (int j = 0; j < intArray[i].length; j++) {
                System.out.println(intArray[i][j]);
            }
        }
    }
}
```



Thank you ❤️

qa-june-2024-automation-with-java-slides